CUNY SPS DATA 621 - CTG5 - HW5

Betsy Rosalen, Gabrielle Bartomeo, Jeremy O'Brien, Lidiia Tronina, Rose Koh

May 15th, 2019

Contents

Table 1: Data Dictionary

VARIABLE	DEFINITION	TYPE
TARGET	Number of Cases Purchased	count response
AcidIndex	Method of testing total acidity by using a weighted avg	continuous numerical predictor
Alcohol	Alcohol Content	continuous numerical predictor
Chlorides	Chloride content of wine	continuous numerical predictor
CitricAcid	Citric Acid Content	continuous numerical predictor
Density	Density of Wine	continuous numerical predictor
FixedAcidity	Fixed Acidity of Wine	continuous numerical predictor
FreeSulfurDioxide	Sulfur Dioxide content of wine	continuous numerical predictor
LabelAppeal	Marketing Score indicating the appeal of label design	categorical predictor
ResidualSugar	Residual Sugar of wine	continuous numerical predictor
STARS	Wine rating by a team of experts. 4 = Excellent, 1 = Poor	categorical predictor
Sulphates	Sulfate conten of wine	continuous numerical predictor
TotalSulfurDioxide	Total Sulfur Dioxide of Wine	continuous numerical predictor
VolatileAcidity	Volatile Acid content of wine	continuous numerical predictor
pH	pH of wine	continuous numerical predictor

1 DATA EXPLORATION

When dining in a restaurant, a sommelier can assist you in selecting a perfect wine, even if you do not know much about wine yourself. By asking about your taste preferences, they can recommend a wine that pairs well with your meal, while complementing your likes and dislikes. But what happens when you're a large wine manufacturer, wondering how to produce wines that will sell? If the wine manufacturer can predict the number of cases sold based on the characteristics of a wine, then that manufacturer will be able to adjust their wine offering to maximize sales.

Our data set contains information on approximately 12,000 commercially available wines. Most of the variables are related to the chemical properties of the wine. The response variable is the number of sample cases that were purchased by wine distribution companies after sampling the wine. These cases would be used to provide tasting samples to restaurants and wine stores around the United States and promote further sales.

1.1 Summary Statistics

Continuous quantitative and categorical variables were summarized separately for the sake of clarity.

`STARS` and `LabelAppeal` can be described as ordinal categorical variables. However, because of the numerical coding, these variables were imported as if they were quantitative. Since they are ordinal consideration was given to treating them as numerical, but ultiately the decision was made to convert them to factors.

1.1.1 Summary Statistics Graphs

The histograms in Figure 1 shows the distribution of all 15 variables. The distribution is more peaked than a normal bell curve for most of our varibales especially `Chlorides`, `CitricAcid`, `Density`, `FixedAcidity` , `FreeSulfurDioxide`, `pH`, `ResidualSugar`, `Sulphates`, `TotalSulfurDioxide` and `ViolatileAcidity`. Most of those are also centered at or near zero. `AcidIndex` has a slight right skew.

Table 2: Summary statistics for numeric variables

	n	min	mean	median	max	sd
AcidIndex	12795	4.00	7.77	8.00	17.0	1.32
Alcohol	12142	-4.70	10.49	10.40	26.5	3.73
Density	12795	0.89	0.99	0.99	1.1	0.03
Sulphates	11585	-3.13	0.53	0.50	4.2	0.93
pH	12400	0.48	3.21	3.20	6.1	0.68
TotalSulfurDioxide	12113	-823.00	120.71	123.00	1057.0	231.91
FreeSulfurDioxide	12148	-555.00	30.85	30.00	623.0	148.71
Chlorides	12157	-1.17	0.05	0.05	1.4	0.32
ResidualSugar	12179	-127.80	5.42	3.90	141.2	33.75
CitricAcid	12795	-3.24	0.31	0.31	3.9	0.86
VolatileAcidity	12795	-2.79	0.32	0.28	3.7	0.78
FixedAcidity	12795	-18.10	7.08	6.90	34.4	6.32
TARGET	12795	0.00	3.03	3.00	8.0	1.93

Table 3: Summary statistics for categorical variables

	STARS	LabelAppeal
	1 :3042	-2: 504
	2 :3570	-1:3136
	3 :2212	0 :5617
	4 : 612	1 :3048
	NA's:3359	2 : 490

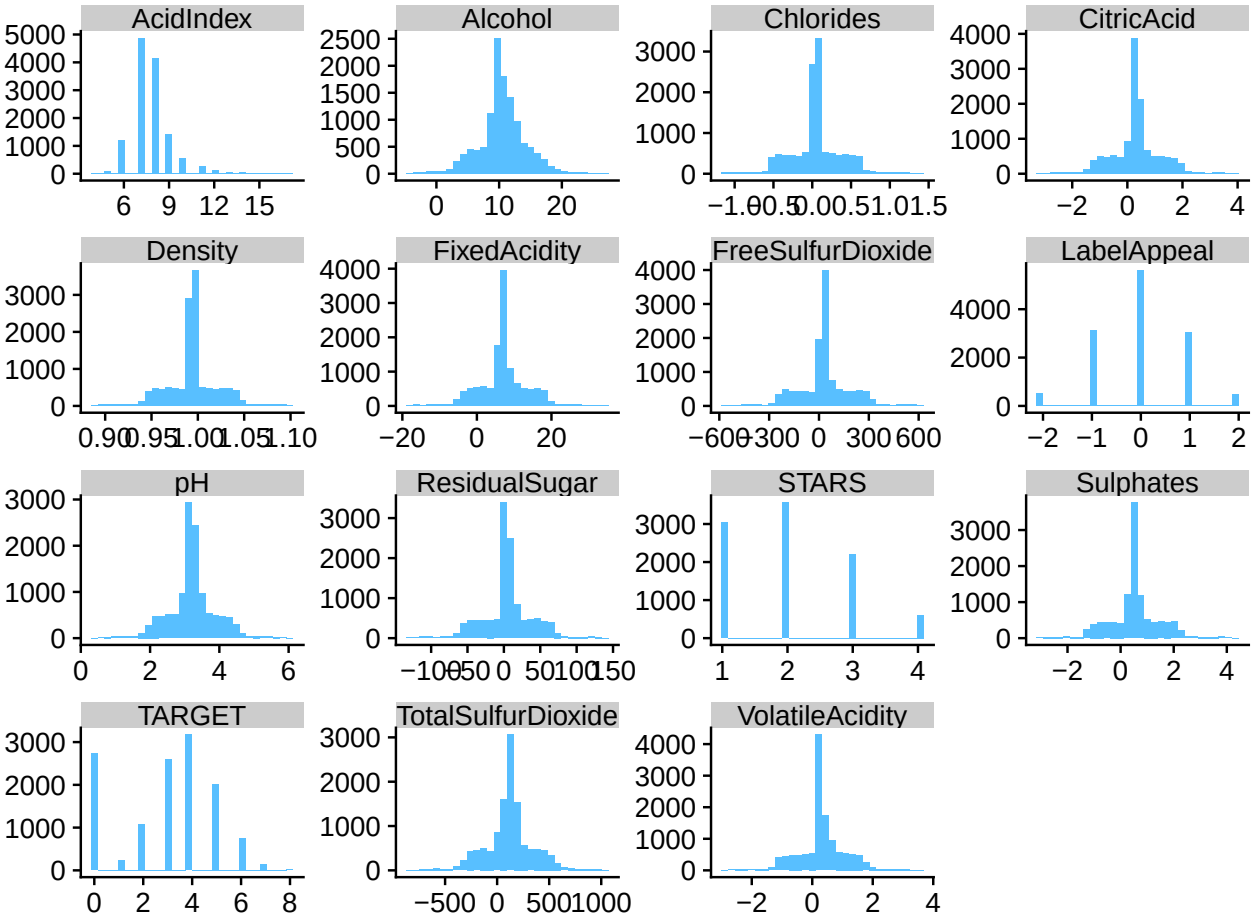


Figure 1: Data Distributions

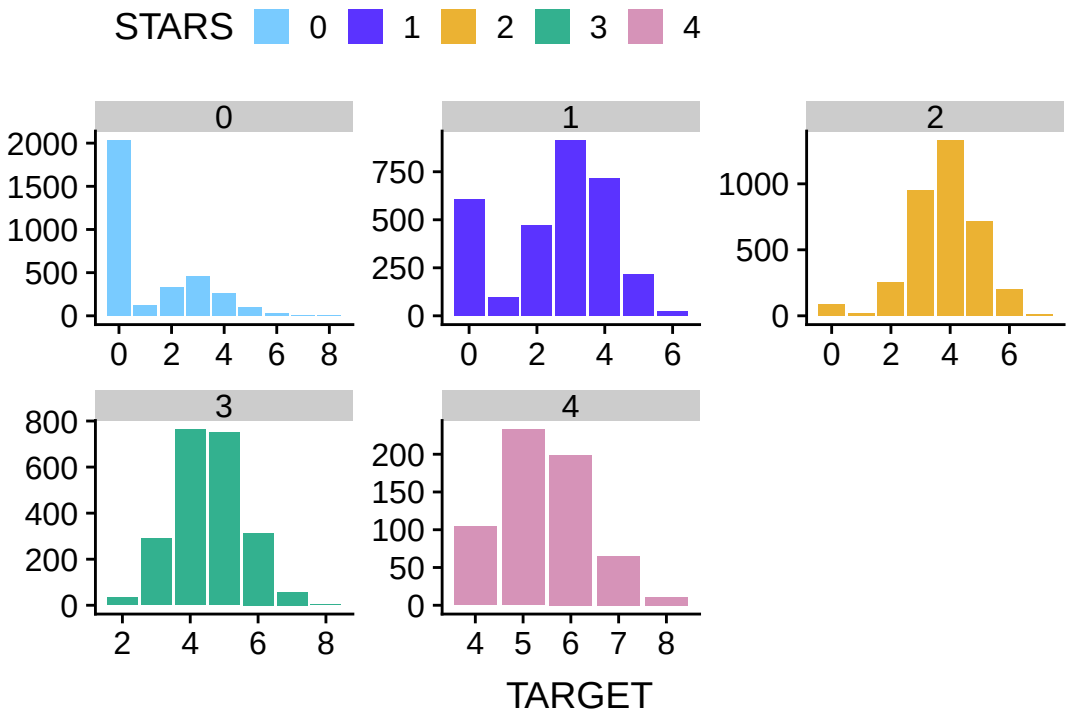


Figure 2: TARGET Distributions by STARS Values

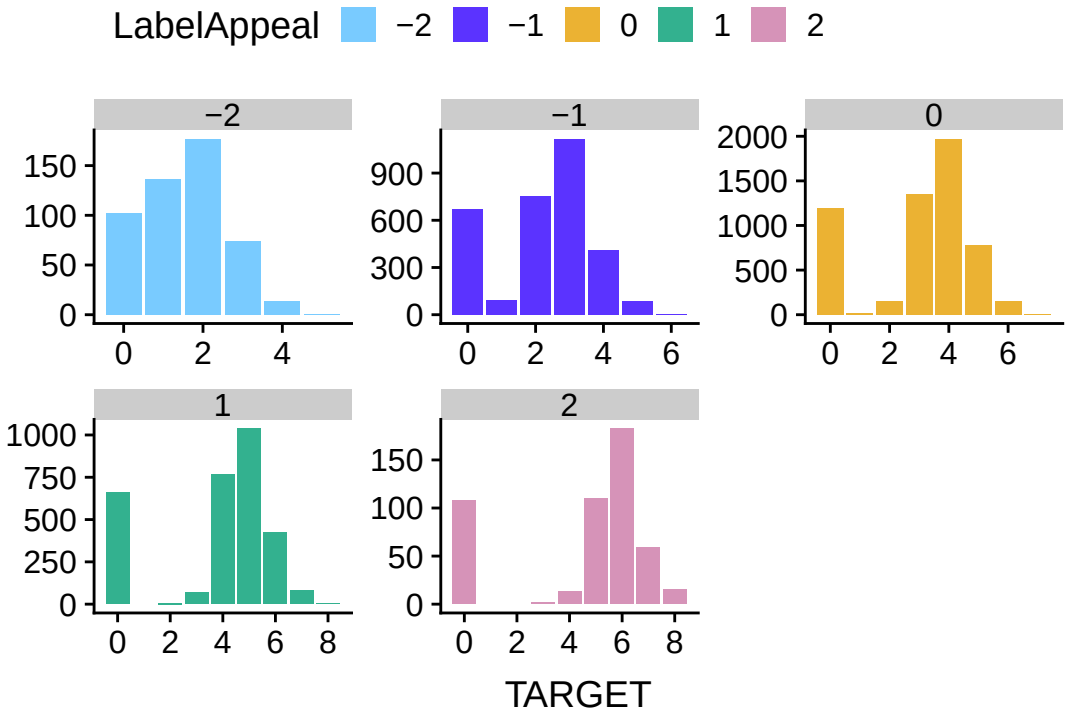


Figure 3: TARGET Distributions by LabelAppeal Values

Figure 2 shows that there are a large number of outliers that need to be accounted for, except for `LabelAppeal`, `AcidIndex` and `STARS` which have limited number of variations.

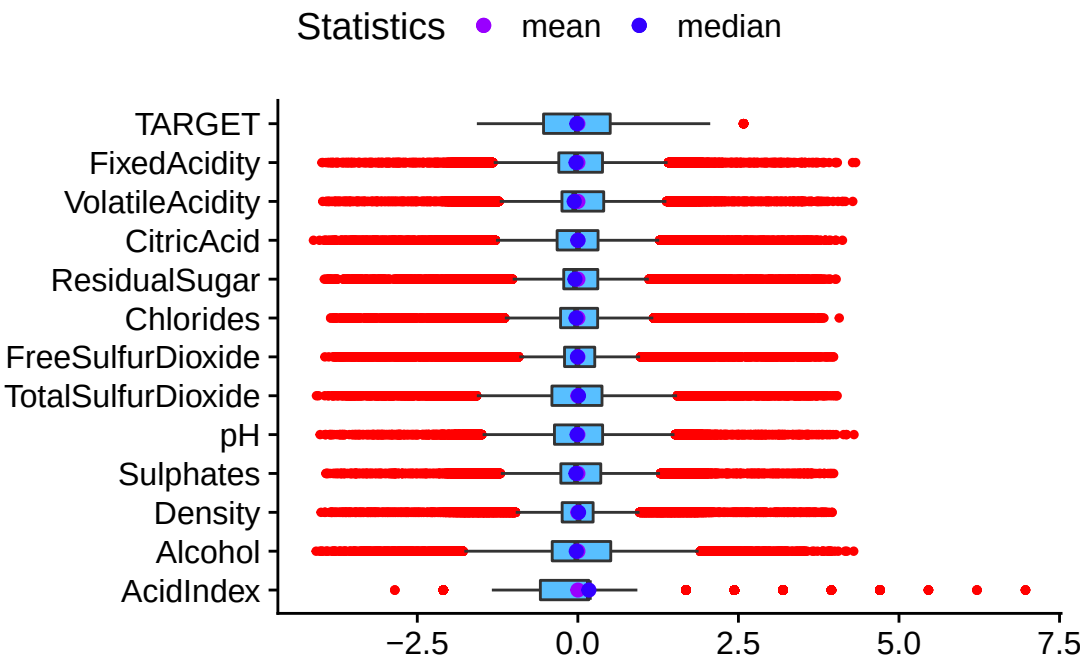


Figure 4: Scaled Boxplots

1.2 Linearity

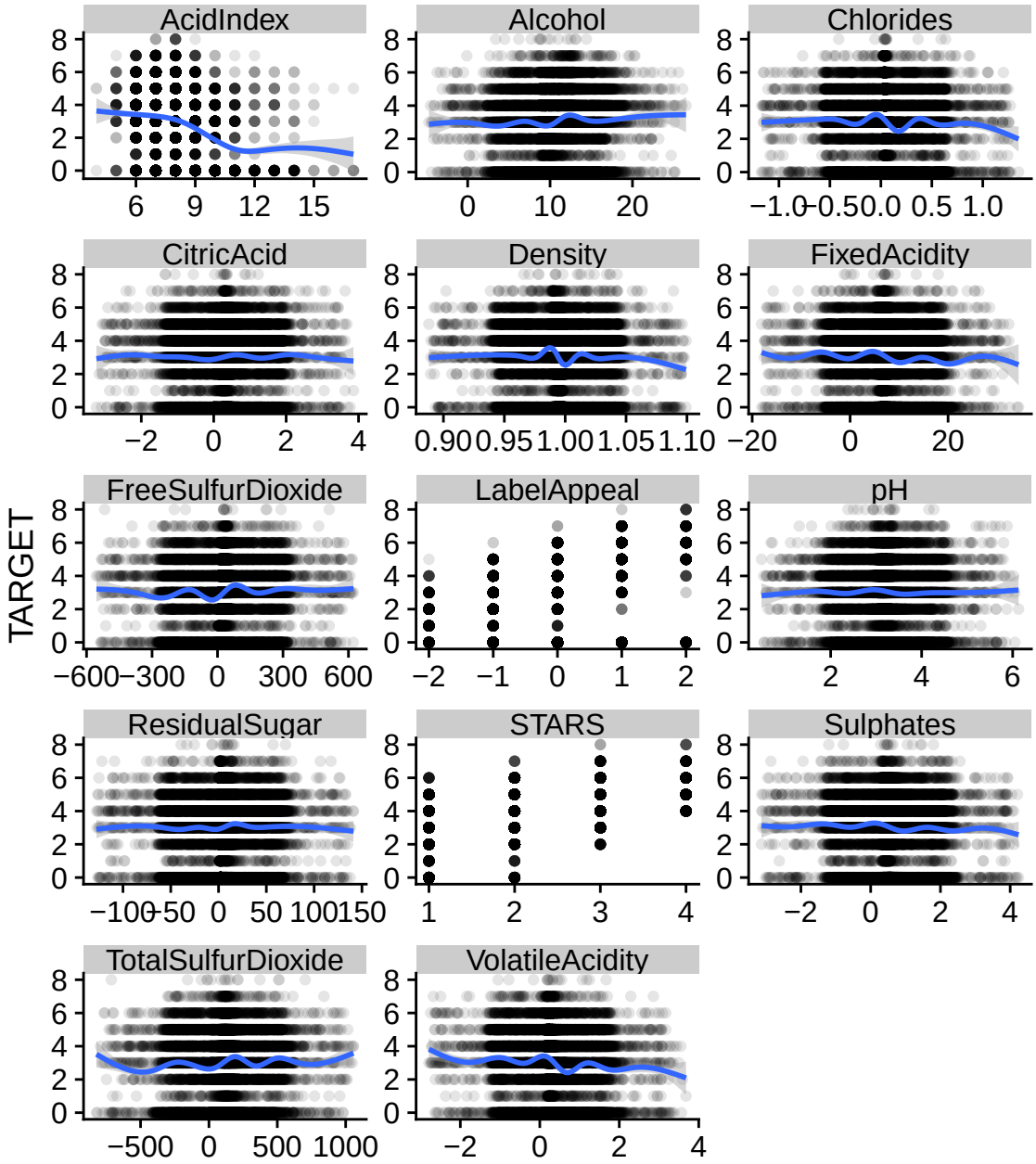


Figure 5: Scatter plot between numeric predictors and the TARGET

The raw predictors fail to show linear relationship with the `TARGET` except for the `AcidIndex` and `VolativeAcidity`. The Scatter Plots show a systematic, wave-like pattern for `Density`, `FixedAcidity`, `FreeSulfurDioxide`, `TotalSulfurDioxide` and `VolativeAcidity`.

1.2.1 Log Transformed Data

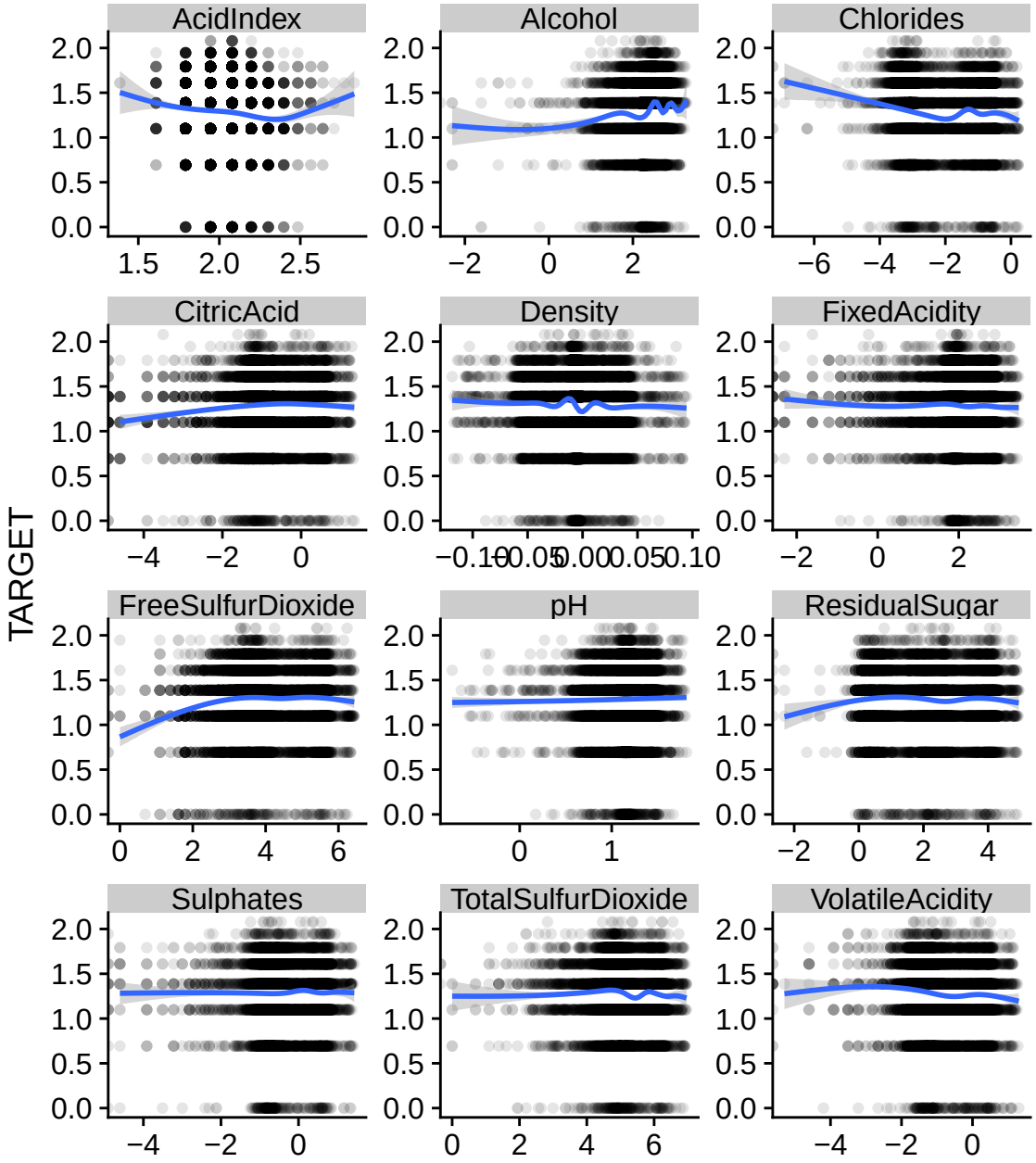


Figure 6: Scatter plot between log transformed predictors and the log transformed TARGET filtered for rows where TARGET is greater than 0

In attempt to improve the linearity of the variables against the `TARGET` variable, we start with a log transformation on all predictors and`TARGET` variable. As a result, the linearity of `Chlorides` and `FreeSulfurDioxide` become more apparent.

1.2.2 Box-Cox

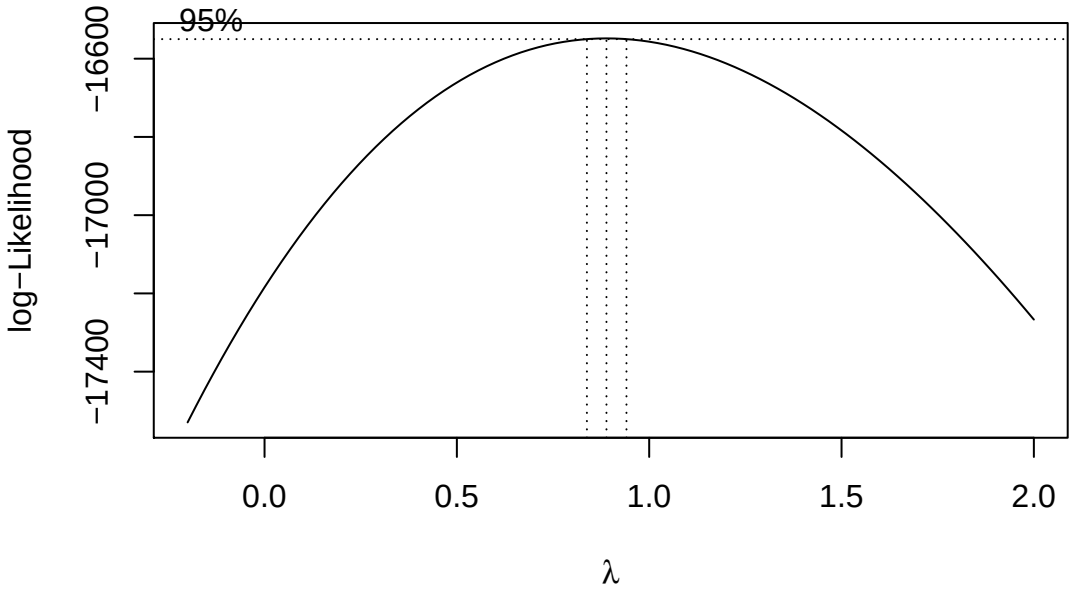


Figure 7: Box-Cox Plot

1.2.3 Square Root Transformed Predictors and Log Transformed Target

In 'Linear Models with R', Faraway suggested that the square root transformation is often appropriate for count response data. The Poisson distribution is a good model for counts, and that distribution has the property that the mean is equal to the variance thus suggesting the square root transformation. A plot of each predictor square root transformed plotted against the log transformed `TARGET`.

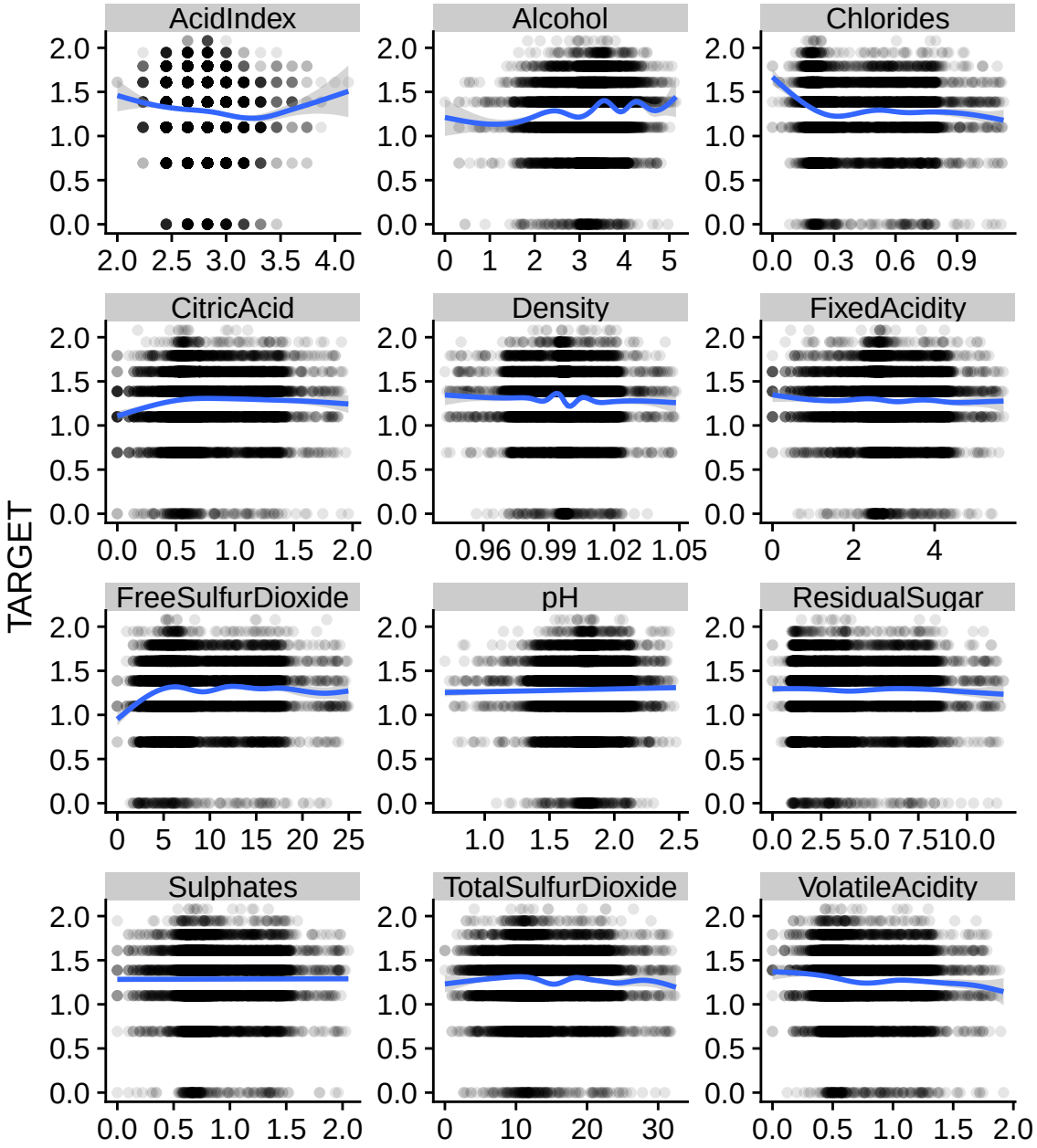


Figure 8: Scatter plot between square root transformed predictors and the square root transformed TARGET filtered for rows where TARGET is greater than 0

1.3 Missing Data

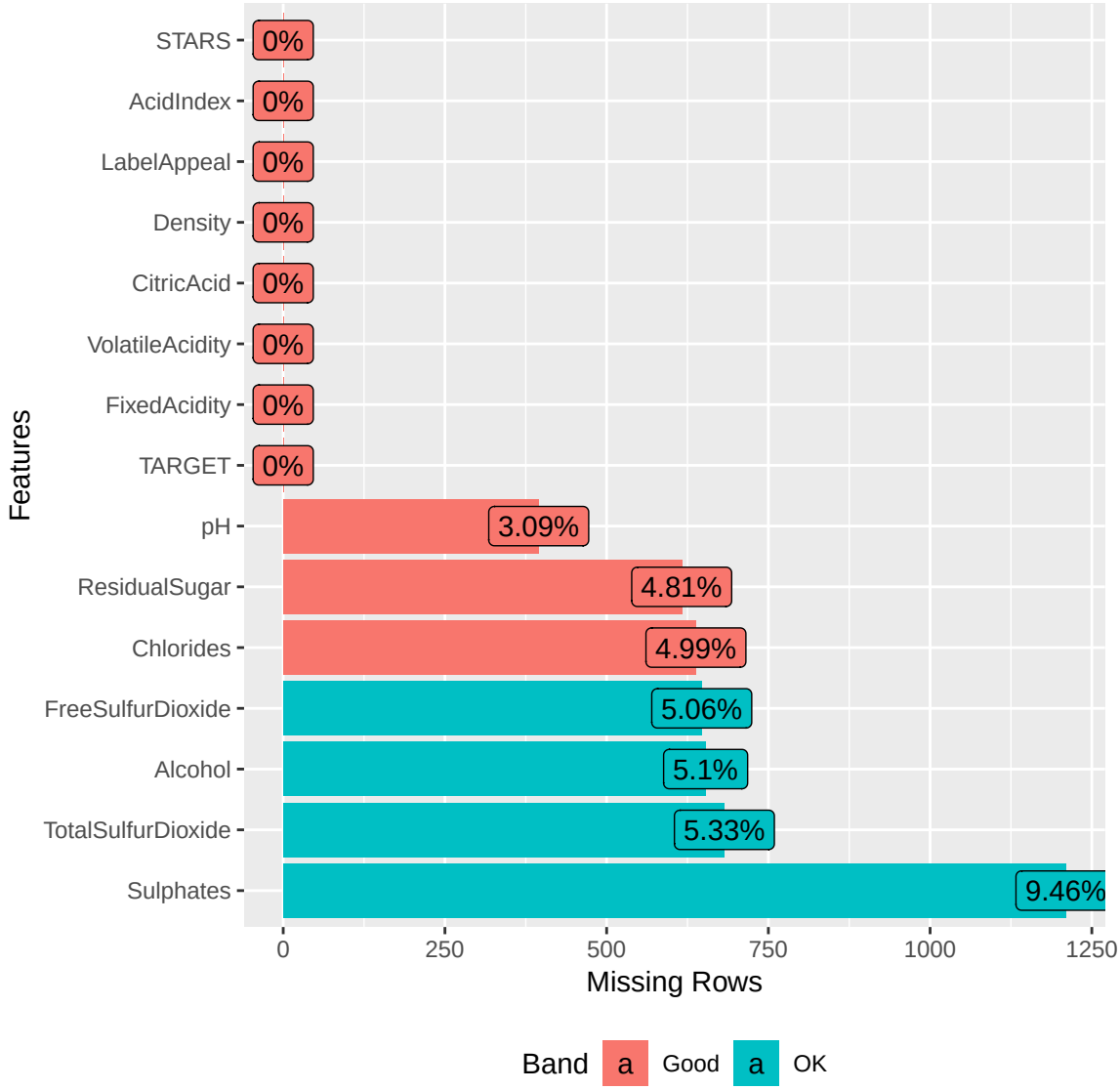


Figure 9: Missing data

A number of variables are missing observations: `STARS`, `Sulphates`, `TotalSulfurDioxide`, `Alcohol`, `FreeSulfurDioxide`, `Chlorides`, `ResidualSugar`, `pH`. For `STARS`, the number is 26.25%, but the others range between 3% and 9% of total. Approximately 50% of the cases are missing one of these variables.

2 DATA PREPARATION

Describe how you have transformed the data by changing the original variables or creating new variables

a. Fix missing values (maybe with a Mean or Median value)

b. Create flags to suggest if a variable was missing

c. Transform data by putting it into buckets

d. Mathematical transforms such as log or square root (or use Box-Cox)

e. Combine variables (such as ratios or adding or multiplying) to create new variables

[JO: Consider strategies to transform negative variables - arithmetic?]

2.1 Missing Values

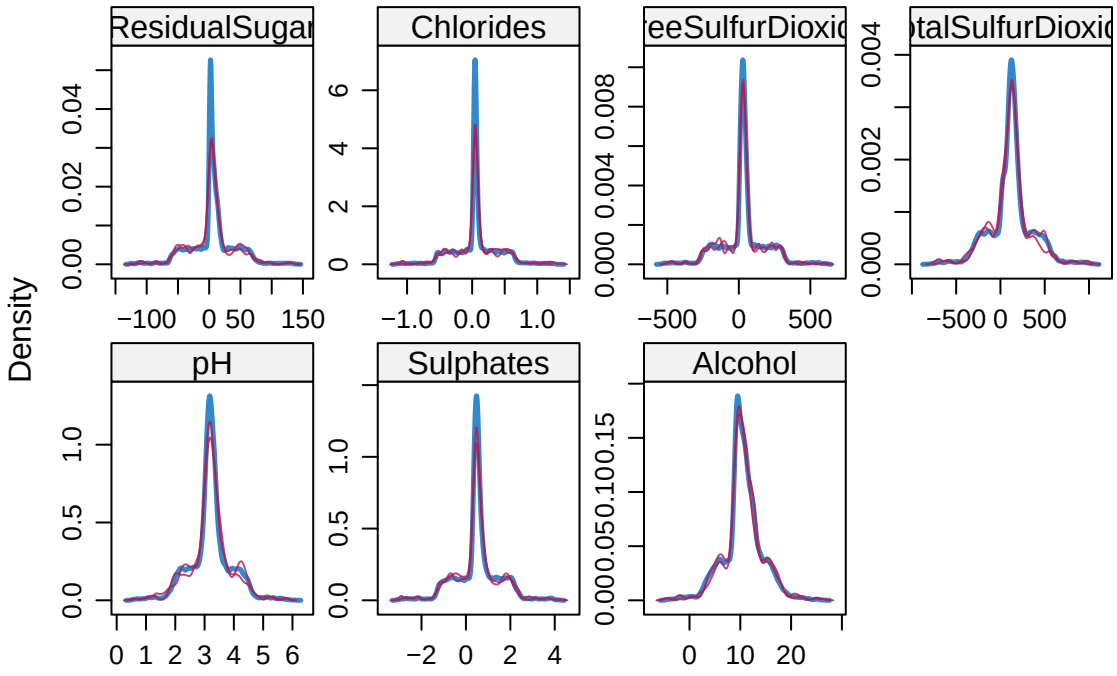


Figure 10: Difference between original and imputed data

There are many observations that are missing multiple pieces of data. Especially the `STARS` miss more than a quarter in both train and test dataset. This does not appear to be missing random pattern thus imputation would not be suitable. Moreover, it appears to be a highly significant predictor. To manage this, we simply assign 0 to NA values in `STARS`. The follwing variables are imputed using MICE package: `Sulphates`, `TotalSulfurDioxide`, `Alcohol`, `FreeSulfurDioxide`, `Chlorides`, `ResidualSugar`, or `pH`.

Pink and blue lines indicate close fit match in distribution of imputed values and recorded values, with the exception of STARS - the addition of STARS values has given the appearance of shifting the distribution from high to low lambda values.

2.2 Transformation / Feature Engineering

There are a large number of negative values for variables for which that is nonsensical, examples: `Alcohol`, `CitricAcid`, `FixedAcidity`, `FreeSulfurDioxide`, `ResidualSugar`, `Sulphates`, `TotalSulfurDioxide`. and `VolatileAcidity`. The range for the Poisson and negative binomial distribution has zero as a lower bound, so we can arithmetically transform the aforementioned variables to scale the lower IQR non-outlier values from zero up and drop the sub-IQR values (now the only negative values remaining). [JO currently function not working as intended - moving all points up by the mean - IQR * 1.5, and tossing anything less than that - so tuning calculations]

(https://www.imachordata.com/do-not-log-transform-count-data-bitches/)

Alternatively, we can explore whether more information on how measurements were made can be found to discern whether there might be some reason for negative values, and if there's a possibility of systematic data errors

We'll also test the data for over-dispersion when setting up negative binomial models.

Table 4: Comparison of different data prep methods influence on AIC

	df	AIC
imputed only	21	45620
plus min	21	45620
plus iqr1.5	21	45609
abs log	21	45636

Table 5: Comparison of different data prep methods influence on model coefficients

	imputed only	plus min	plus iqr1.5	abs log	range
(Intercept)	0.69	0.76	0.70	1.44	0.75
FixedAcidity	0.00	0.00	-0.27	-0.57	0.57
VolatileAcidity	-0.03	-0.03	-0.01	-0.04	0.03
CitricAcid	0.01	0.01	0.00	-0.67	0.68
ResidualSugar	0.00	0.00	0.23	0.00	0.23
Chlorides	-0.04	-0.04	0.42	-0.08	0.50
FreeSulfurDioxide	0.00	0.00	0.56	0.03	0.56
TotalSulfurDioxide	0.00	0.00	0.69	0.01	0.69
Density	-0.26	-0.26	-0.08	-0.05	0.21
pH	-0.01	-0.01	0.77	0.02	0.78
Sulphates	-0.01	-0.01	1.08	0.03	1.09
Alcohol	0.00	0.00	1.20	-0.03	1.24
LabelAppeal-1	0.24	0.24	1.32	0.03	1.29
LabelAppeal0	0.43	0.43	0.00	0.24	0.43
LabelAppeal1	0.56	0.56	-0.03	0.43	0.58
LabelAppeal2	0.70	0.70	0.01	0.56	0.69
AcidIndex	-0.08	-0.08	0.00	0.69	0.77
STARS1	0.77	0.77	-0.03	0.77	0.80
STARS2	1.09	1.09	0.00	1.09	1.09
STARS3	1.20	1.20	0.00	1.21	1.21
STARS4	1.33	1.33	-0.01	1.33	1.34

3 BUILD MODELS

`Using the training data set, build at least two different poisson regression models, at least two diffe`

`Discuss the coefficients in the models, do they make sense? In this case, about the only thing you can`

3.1 Poisson Regression Model

Poisson regression models were run on all 4 data preparation methods we chose using a full model approach that included all predictors in each model. The data preparation method did not seem to have any meaningful impact on the performance of the models with all four resulting in similar AIC values although the coefficients were significantly different.

Further models were created using the data prep strategy that involved adding the minimum value plus one to all variables that had negative values in order to remove all negative and zero values from our data.

Using the full model above as a base a `step` function was run to reduce the number of variables in the model resulting in signifcantly fewer varibles. The drop1 function was then run to see if any more variables could be reomved and `pH` was dropped from tjhe model as a result.

An influence plot was also run to see if there were any influential points affecting the mdoel. Five influential points (records 3953, 4940, 8887, 10108, and 12513) were identified and removed from the model to see if they improved the fit. The improvement was modest but noticable so they were left out of the remaining poisson models.

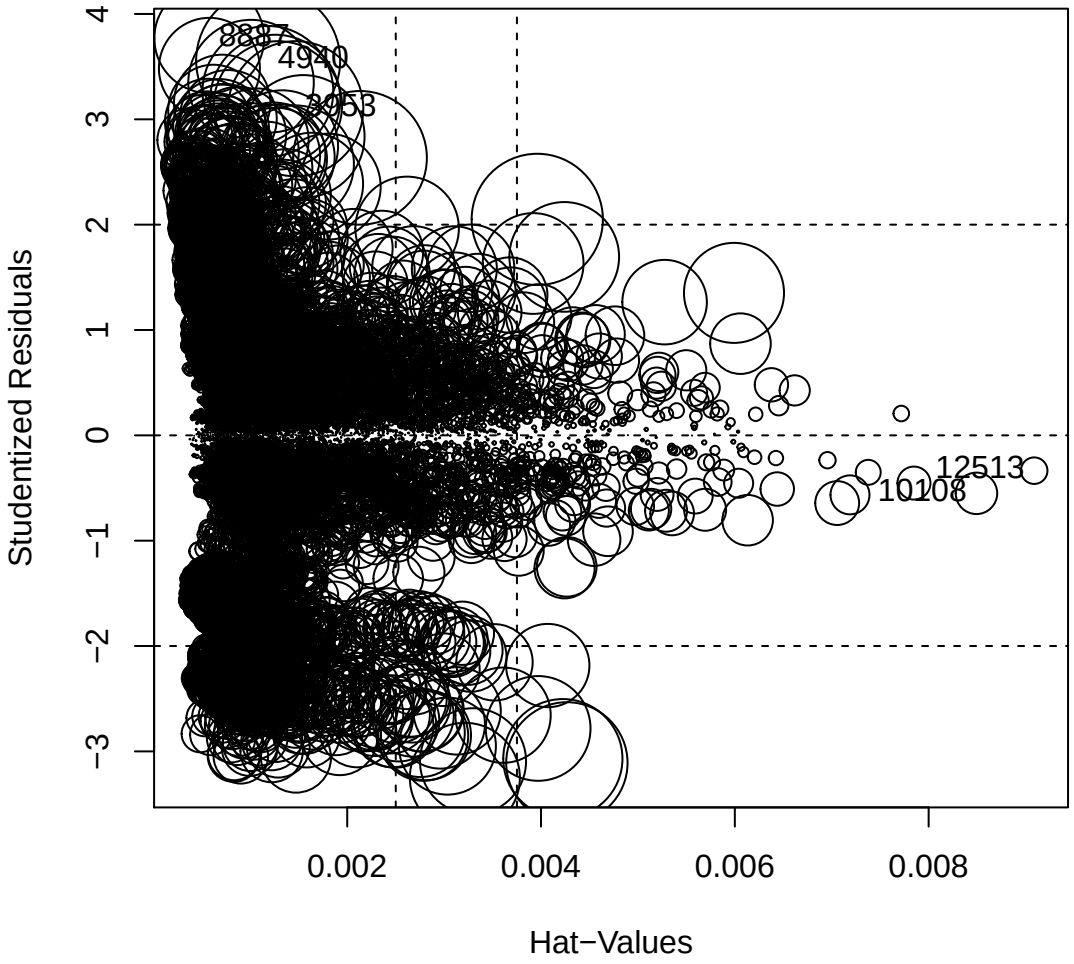


Figure 11: Poisson model influence plot

StudRes	Hat	CookD
3.1	0.00147	0.00161
3.56	0.00118	0.00183
3.77	0.000576	0.00106
-0.551	0.00849	0.000152
-0.335	0.00909	6.2e-05

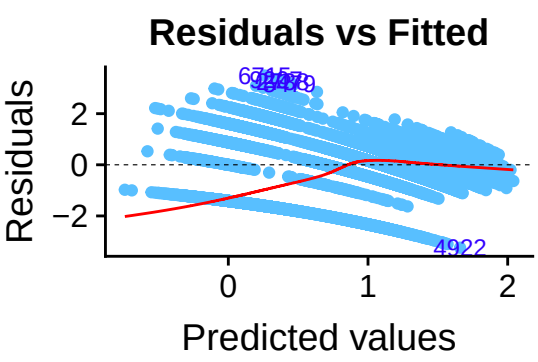
3.1.0.1 Base Poisson Model Statistics and Coefficients

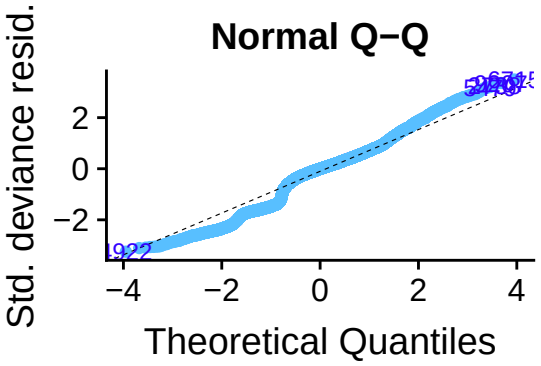
Observations	12790
Dependent variable	TARGET
Type	Generalized linear model
Family	poisson
Link	log

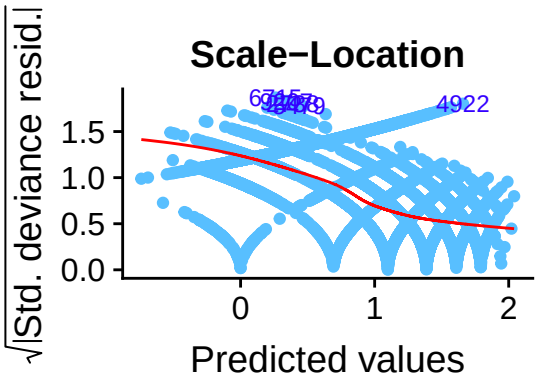
$\chi^2(15)$	9235.52
Pseudo-R² (Cragg-Uhler)	0.52
Pseudo-R² (McFadden)	0.17
AIC	45559.21
BIC	45678.52

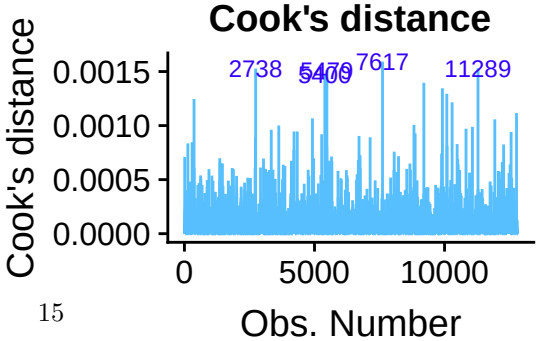
	Est.	S.E.	z val.	p	VIF
(Intercept)	0.49	0.08	5.85	0.00	NA
STARS1	0.77	0.02	39.40	0.00	1.17
STARS2	1.09	0.02	59.71	0.00	1.17
STARS3	1.21	0.02	62.93	0.00	1.17
STARS4	1.33	0.02	54.70	0.00	1.17
LabelAppeal-1	0.24	0.04	6.19	0.00	1.13
LabelAppeal0	0.42	0.04	11.47	0.00	1.13
LabelAppeal1	0.56	0.04	14.76	0.00	1.13
LabelAppeal2	0.69	0.04	16.22	0.00	1.13
AcidIndex	-0.08	0.00	-17.58	0.00	1.03
VolatileAcidity	-0.03	0.01	-4.82	0.00	1.00
TotalSulfurDioxide	0.00	0.00	3.55	0.00	1.00
Alcohol	0.00	0.00	2.83	0.00	1.01
FreeSulfurDioxide	0.00	0.00	2.67	0.01	1.00
Sulphates	-0.01	0.01	-1.89	0.06	1.00
Chlorides	-0.04	0.02	-2.24	0.03	1.00

Standard errors: MLE









3.1.1 Quasipoisson Model

Next a quasipoisson model was tried using all the same steps outlined for the poisson model above that resulted in a model that had very little change and no real improvement to the model fit over the poisson model. The standard errors of the coefficients were moderately reduced for about half the variables but significantly increased for the other half.

poisson	quasip	se.poiss	se.quasi	ratio
0.486	0.486	0.0831	0.0779	0.938
0.771	-0.0315	0.0196	0.0061	0.313
1.09	-0.0358	0.0183	0.015	0.82
1.21	0.0001	0.0192	0	0.0017
1.33	0.0001	0.0244	0	0.0009
0.235	-0.0103	0.038	0.0051	0.135
0.425	0.0039	0.0371	0.0013	0.0348
0.556	0.235	0.0377	0.0356	0.945
0.689	0.425	0.0425	0.0347	0.818
-0.0792	0.556	0.0045	0.0353	7.85
-0.0315	0.689	0.0065	0.0398	6.09
0.0001	-0.0792	0	0.0042	191
0.0039	0.771	0.0014	0.0184	13.3
0.0001	1.09	0	0.0171	500
-0.0103	1.21	0.0054	0.018	3.31
-0.0358	1.33	0.016	0.0228	1.43

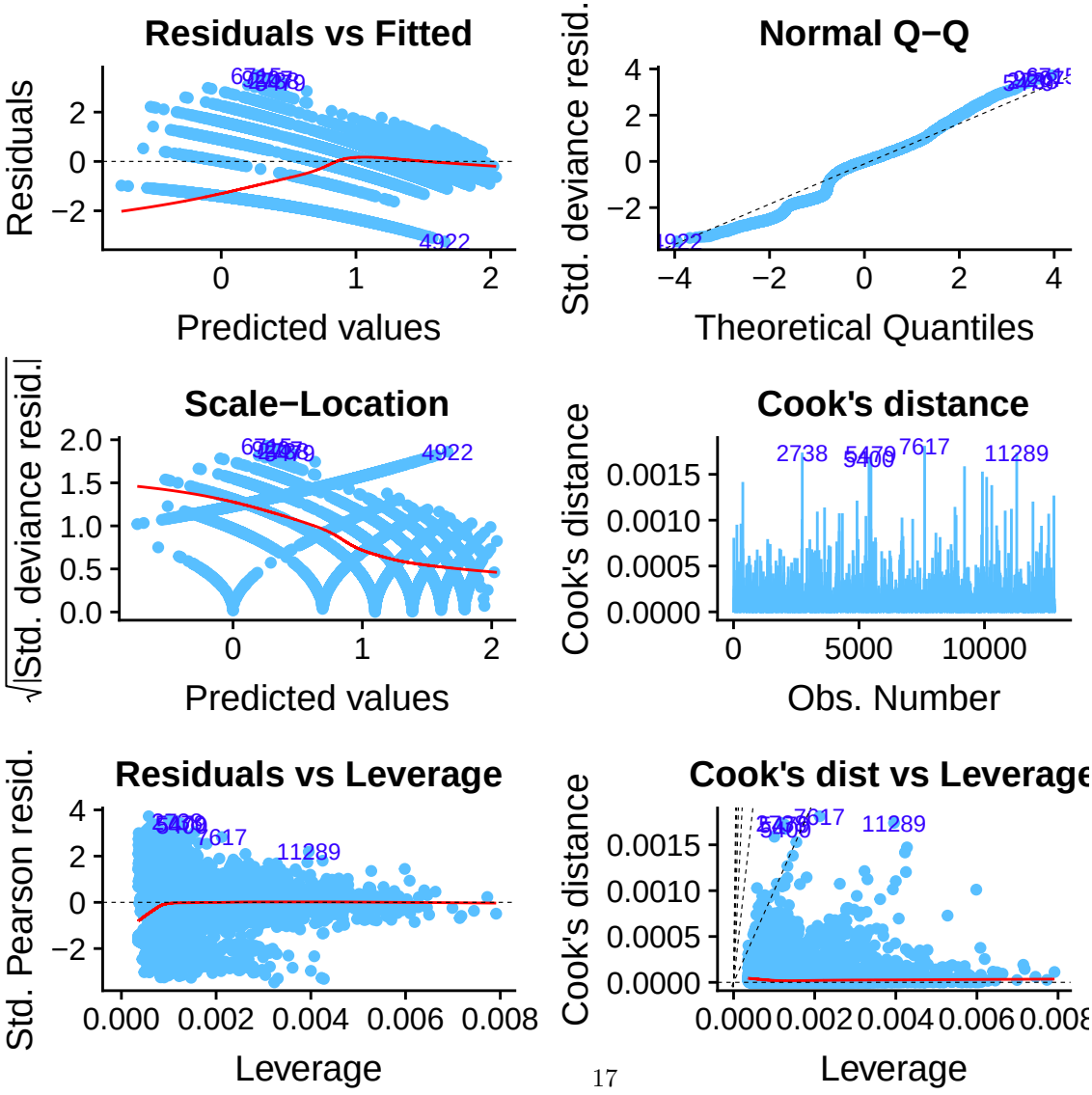
3.1.1.1 Quasipoisson Model Statistics and Coefficients

Observations	12790
Dependent variable	TARGET
Type	Generalized linear model
Family	quasipoisson
Link	log

$\chi^2(15)$	9235.52
Pseudo-R^2 (Cragg-Uhler)	0.52
Pseudo-R^2 (McFadden)	0.17
AIC	NA
BIC	NA

	Est.	S.E.	t val.	p	VIF
(Intercept)	0.49	0.08	6.24	0.00	NA
VolatileAcidity	-0.03	0.01	-5.13	0.00	1.00
Chlorides	-0.04	0.01	-2.39	0.02	1.00
FreeSulfurDioxide	0.00	0.00	2.85	0.00	1.00
TotalSulfurDioxide	0.00	0.00	3.78	0.00	1.00
Sulphates	-0.01	0.01	-2.02	0.04	1.00
Alcohol	0.00	0.00	3.01	0.00	1.01
LabelAppeal-1	0.24	0.04	6.60	0.00	1.13
LabelAppeal0	0.42	0.03	12.23	0.00	1.13
LabelAppeal1	0.56	0.04	15.74	0.00	1.13
LabelAppeal2	0.69	0.04	17.29	0.00	1.13
AcidIndex	-0.08	0.00	-18.74	0.00	1.03
STARS1	0.77	0.02	42.02	0.00	1.17
STARS2	1.09	0.02	63.67	0.00	1.17
STARS3	1.21	0.02	67.11	0.00	1.17
STARS4	1.33	0.02	58.33	0.00	1.17

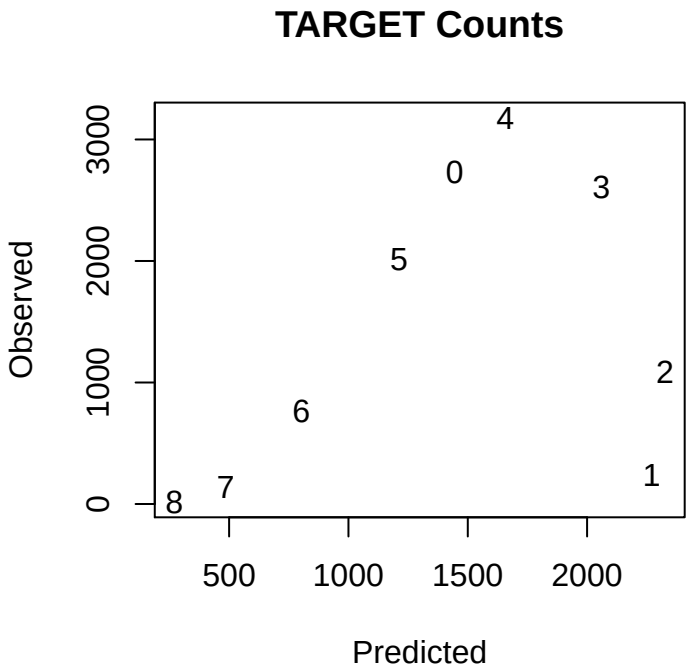
Standard errors: MLE



3.1.2 Hurdle and Zero-inflated Poisson Models

Hurdle and Zero-inflated models are two ways of dealing with zero-inflated data. a A plot showing the expected vs. observed counts of the `TARGET` values was created in order to see if the data might be zero-inflated. We can clearly see in the plot that we have a much greater occurance of zero values than we would expect while the rest of the values are much closer to our expectations. The shape of the plot does peak at 3, 4 and 5 crases and then curve downward on the right side at 1, 2, and 3 cases indicating that the observed values for the purchase of 3, 4 or 5 cases are much higher than we would expect and the observed values for the purchase of 1 or 2 cases are much lower than we would expect.

```
## integer(0)
```



Plots of the two categorical variables, `STARS` and `LabelAppeal`, when the target is zero vs. when it is not zero show a sginifcant difference for `STARS` but virtually no difference for `LabelAppeal`. The `STARS` plot strongly indicates that a hurdle or zero-inflated or possibly piecewise model may be a better fit. While the `LabelAppeal` plot only supports the choice of hurdle and zero-inflated models.

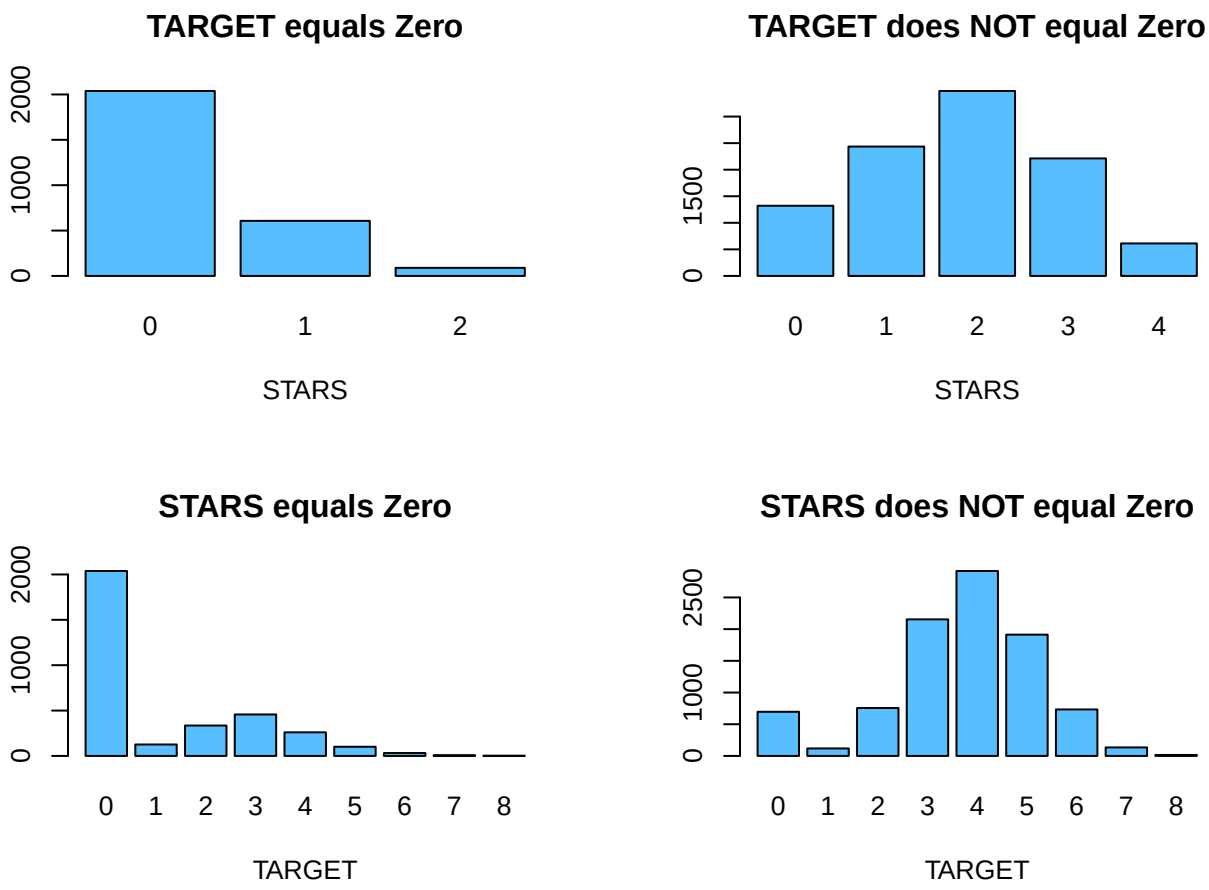


Figure 12: Difference in distributions for STARS when TARGET equals zero vs. greater than zero

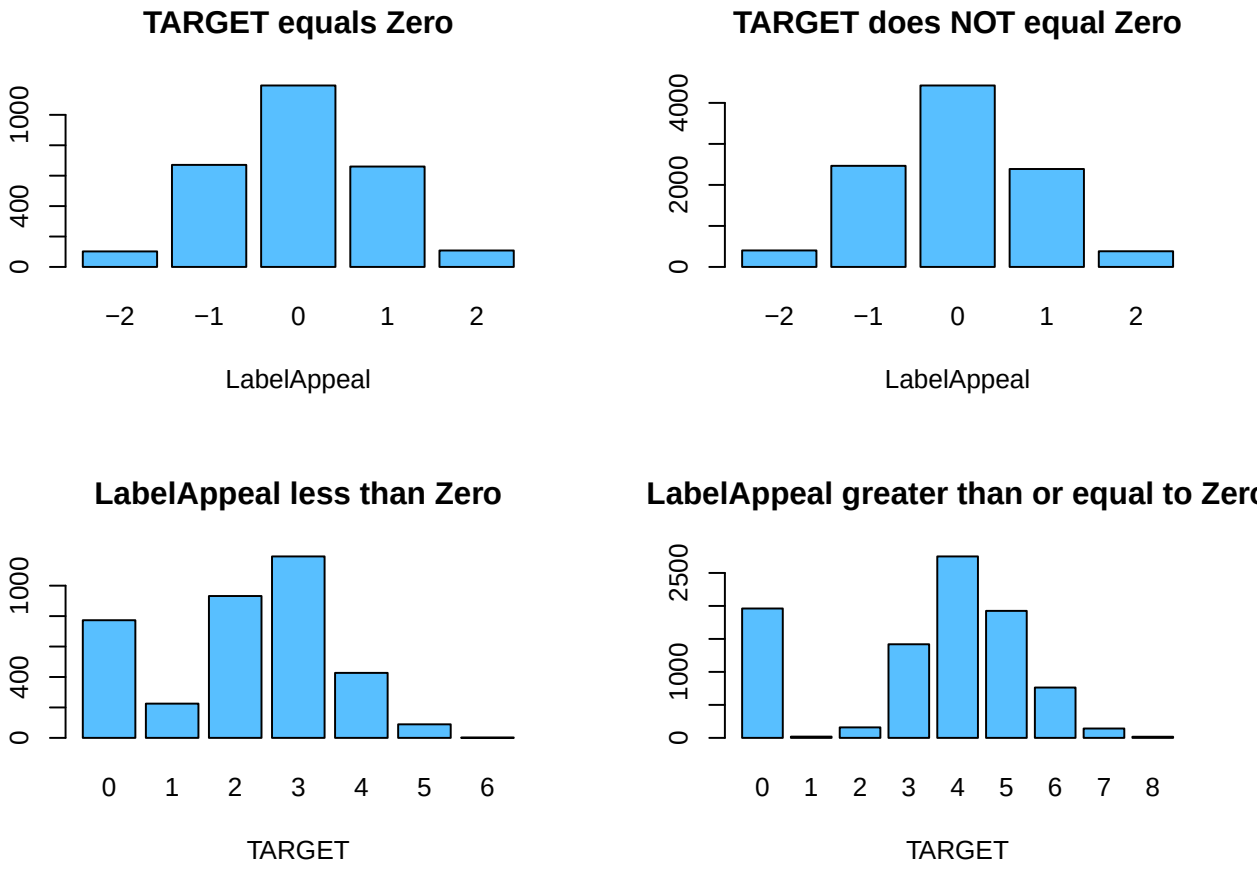
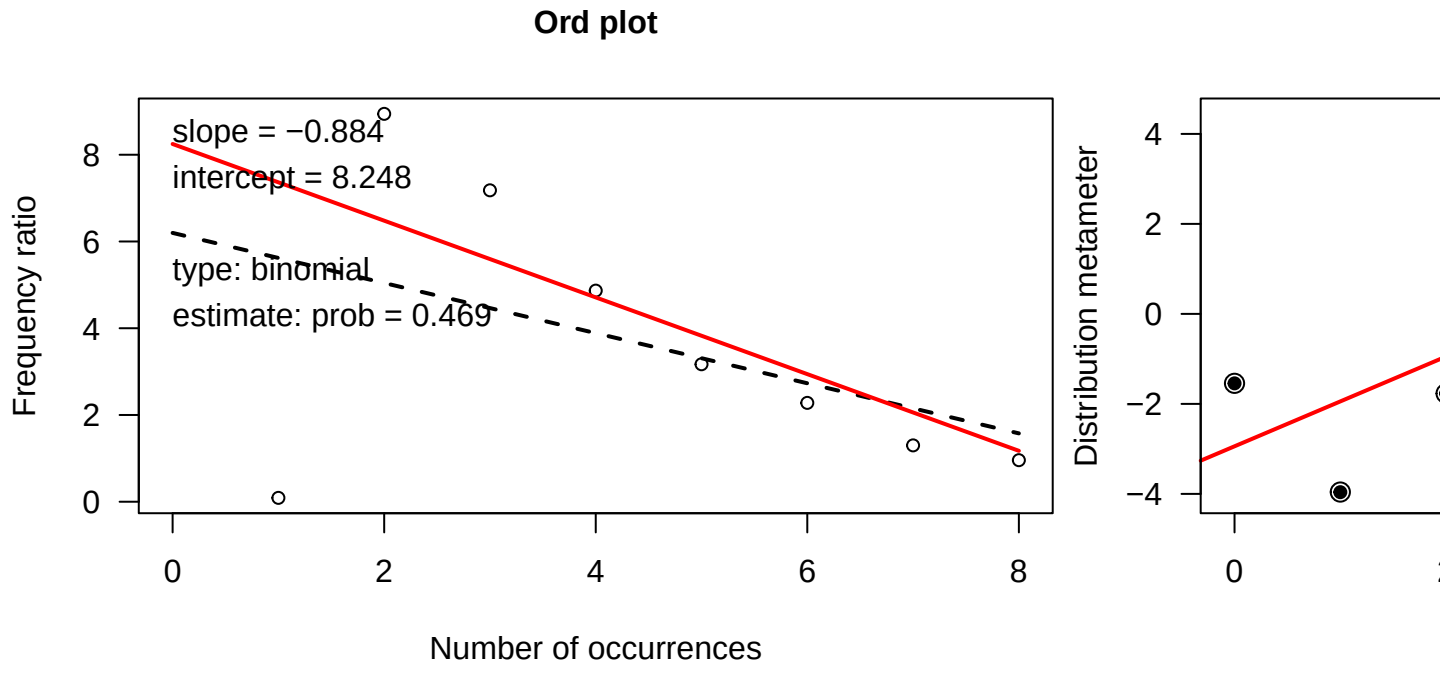
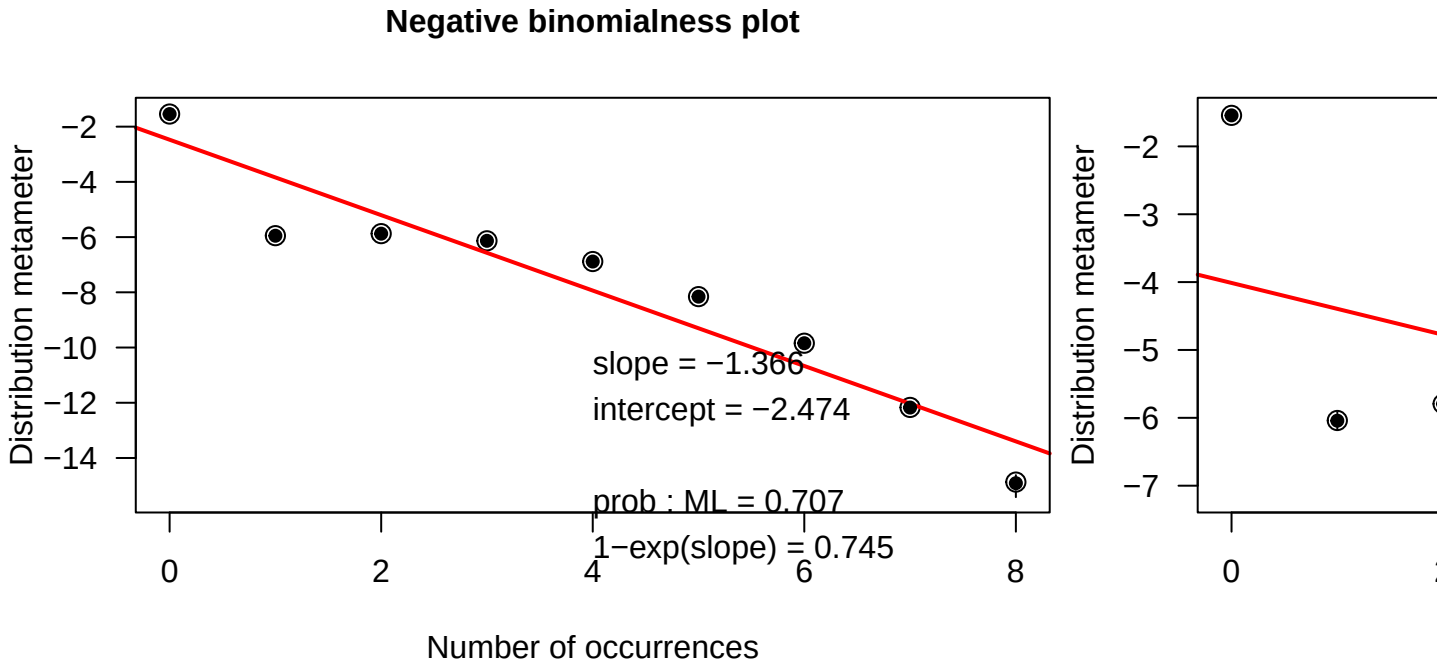


Figure 13: Difference in distributions for LabelAppeal when TARGET is less than or equal to zero vs. greater than zero

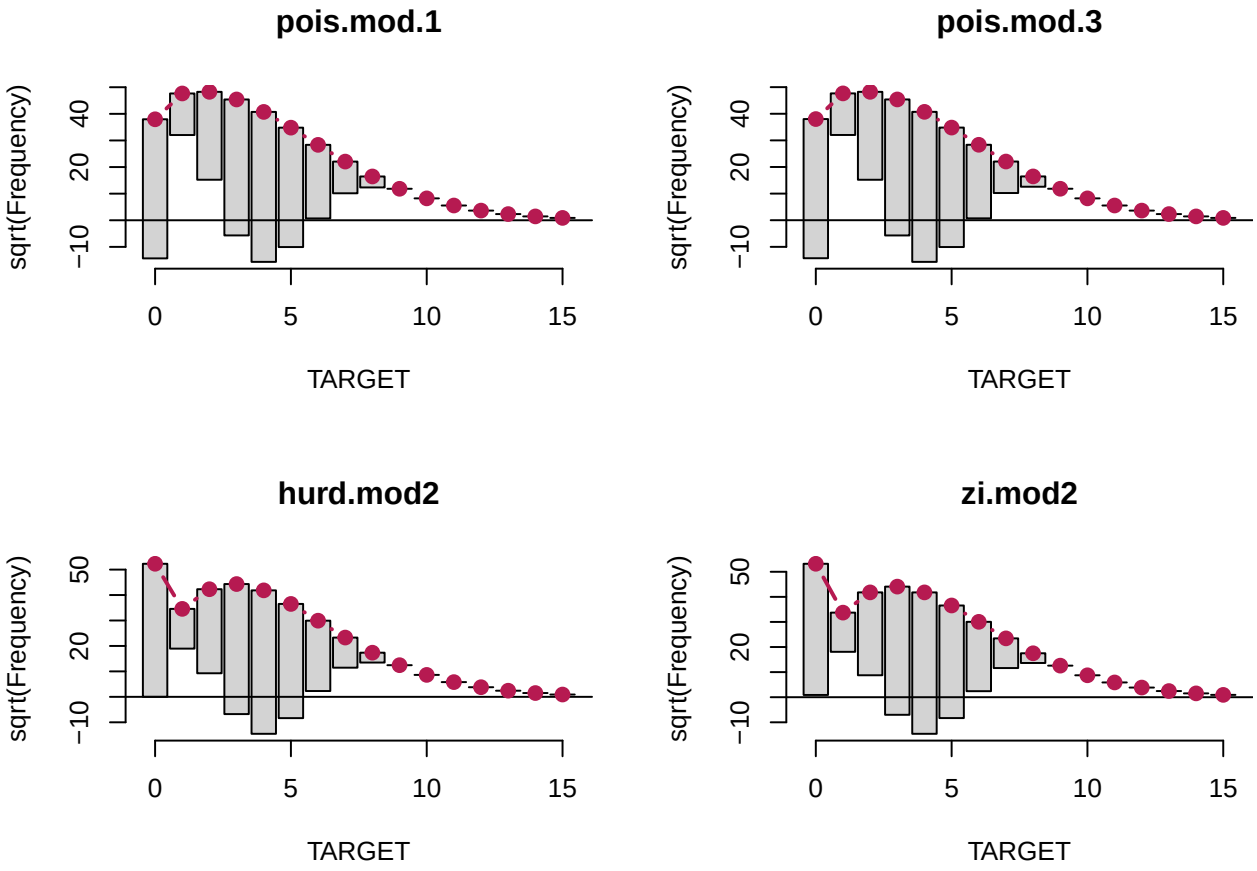
Zero-inflation is also clearly seen in the Diagnostic Distribution Plots below.





Hurdle and zero-inflated models were both run and refined using backward elimination on the data with the influential points already removed. Both models showed significant improvement over the poisson and quasipoisson models with significantly lower AIC values and rootogram plots that showed a closer fit and less dispersion, although dispersion is still an issue since there is still evidence of over and under dispersion in the hurdle and zero-inflated rootogram plots.

3.1.2.1 Rootogram Plots



3.1.2.2 Hurdle Model Statistics and Coefficients

```
##
## Call:
## hurdle(formula = TARGET ~ AcidIndex + Alcohol + LabelAppeal + STARS |
##     VolatileAcidity + FreeSulfurDioxide + TotalSulfurDioxide + pH +
##         Sulphates + Alcohol + LabelAppeal + AcidIndex + STARS, data = minusinfluential)
##
## Pearson residuals:
##      Min       1Q   Median       3Q      Max
## -2.09275 -0.44331 -0.00245  0.39568  4.54188
##
## Count model coefficients (truncated poisson with log link):
##               Estimate Std. Error z value             Pr(>|z|)
## (Intercept)    0.33863    0.06713    5.04           0.00000046 ***
## AcidIndex     -0.01705    0.00492   -3.46              0.00053 ***
## Alcohol        0.00732    0.00145    5.07           0.00000041 ***
## LabelAppeal-1  0.53977    0.04973   10.85 < 0.0000000000000002 ***
## LabelAppeal0   0.84299    0.04880   17.27 < 0.0000000000000002 ***
## LabelAppeal1   1.03971    0.04937   21.06 < 0.0000000000000002 ***
## LabelAppeal2   1.19915    0.05324   22.53 < 0.0000000000000002 ***
## STARS1         0.05342    0.02146    2.49              0.01283 *
## STARS2         0.16856    0.02002    8.42 < 0.0000000000000002 ***
## STARS3         0.25901    0.02098   12.35 < 0.0000000000000002 ***
## STARS4         0.36350    0.02592   14.02 < 0.0000000000000002 ***
## Zero hurdle model coefficients (binomial with logit link):
```

```
##                       Estimate Std. Error z value             Pr(>|z|)
## (Intercept)           4.433024   0.399013   11.11 < 0.0000000000000002 ***
## VolatileAcidity      -0.187381   0.036488   -5.14       0.000000281567 ***
## FreeSulfurDioxide     0.000666   0.000197    3.37              0.00074 ***
## TotalSulfurDioxide    0.000819   0.000127    6.46       0.000000000103 ***
## pH                   -0.172426   0.041928   -4.11       0.000039149953 ***
## Sulphates            -0.095360   0.030502   -3.13              0.00177 **
## Alcohol              -0.018903   0.007666   -2.47              0.01367 *
## LabelAppeal-1        -0.483545   0.137302   -3.52              0.00043 ***
## LabelAppeal0         -0.903071   0.134048   -6.74       0.000000000016 ***
## LabelAppeal1         -1.448751   0.143581  -10.09 < 0.0000000000000002 ***
## LabelAppeal2         -1.879326   0.223186   -8.42 < 0.0000000000000002 ***
## AcidIndex            -0.388161   0.021425  -18.12 < 0.0000000000000002 ***
## STARS1                1.831113   0.061349   29.85 < 0.0000000000000002 ***
## STARS2                4.268926   0.117246   36.41 < 0.0000000000000002 ***
## STARS3               20.250040 363.224872    0.06              0.95554
## STARS4               20.406402 695.238140    0.03              0.97658
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Number of iterations in BFGS optimization: 18
## Log-likelihood: -2.03e+04 on 27 Df
```

3.1.2.3 Zero-inflated Poisson Model Statistics and Coefficients

```
##
## Call:
## zeroinfl(formula = TARGET ~ AcidIndex + Alcohol + LabelAppeal +
##     STARS | VolatileAcidity + FreeSulfurDioxide + TotalSulfurDioxide +
##     pH + Sulphates + Alcohol + LabelAppeal + AcidIndex + STARS,
##     data = minusinfluential)
##
## Pearson residuals:
##      Min       1Q   Median       3Q      Max
## -2.24755 -0.42723  0.00257  0.38363  5.33255
##
## Count model coefficients (poisson with log link):
##               Estimate Std. Error z value             Pr(>|z|)
## (Intercept)    0.47099    0.06056    7.78   0.0000000000000074 ***
## AcidIndex     -0.01960    0.00482   -4.07   0.0000476666666659 ***
## Alcohol        0.00685    0.00140    4.88   0.0000010526444344 ***
## LabelAppeal-1  0.44004    0.04128   10.66 < 0.0000000000000002 ***
## LabelAppeal0   0.72789    0.04035   18.04 < 0.0000000000000002 ***
## LabelAppeal1   0.91746    0.04102   22.36 < 0.0000000000000002 ***
## LabelAppeal2   1.07369    0.04560   23.55 < 0.0000000000000002 ***
## STARS1         0.06468    0.02118    3.05               0.0023 **
## STARS2         0.18706    0.01981    9.44 < 0.0000000000000002 ***
## STARS3         0.28455    0.02075   13.72 < 0.0000000000000002 ***
## STARS4         0.38397    0.02567   14.96 < 0.0000000000000002 ***
##
## Zero-inflation model coefficients (binomial with logit link):
##                       Estimate Std. Error z value             Pr(>|z|)
## (Intercept)          -6.294463   0.569851  -11.05 < 0.0000000000000002 ***
## VolatileAcidity       0.198885   0.043469    4.58      0.0000047553093 ***
```

```
## FreeSulfurDioxide   -0.000818   0.000238   -3.44             0.00058 ***
## TotalSulfurDioxide  -0.000912   0.000152   -5.99      0.0000000021358 ***
## pH                   0.198507   0.049925    3.98      0.0000700470654 ***
## Sulphates            0.116125   0.036433    3.19              0.00144 **
## Alcohol              0.025501   0.009232    2.76              0.00574 **
## LabelAppeal-1        1.497177   0.329672    4.54      0.0000055878285 ***
## LabelAppeal0         2.254987   0.326985    6.90      0.0000000000053 ***
## LabelAppeal1         2.968020   0.332370    8.93 < 0.0000000000000002 ***
## LabelAppeal2         3.492351   0.385069    9.07 < 0.0000000000000002 ***
## AcidIndex            0.430443   0.025686   16.76 < 0.0000000000000002 ***
## STARS1              -2.091442   0.076112  -27.48 < 0.0000000000000002 ***
## STARS2              -5.731503   0.322363  -17.78 < 0.0000000000000002 ***
## STARS3             -20.250172 339.002169   -0.06              0.95237
## STARS4             -20.406439 639.781273   -0.03              0.97456
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Number of iterations in BFGS optimization: 36
## Log-likelihood: -2.03e+04 on 27 Df
```

3.2 Negative binomial regression models

Negative binomial regression is an alternative when there is overdispersion ($var(Y_i) > E(Y_i)$).

"A Poisson distribution is parameterized by λ, which happens to be both its mean and variance. While convenient, it's not often realistic. A distribution of counts will usually have a variance that's not equal to its mean. When we see this happen with data that we assume is Poisson distributed, we say we have under- or overdispersion, depending on if the variance is smaller or larger than the mean. Performing Poisson regression on count data that exhibits this behavior results in a model that doesn't fit well."

"One approach that addresses this issue is the Negative Binomial Regresion. The negative binomial distribution describes the probabilities of the occurrence of whole numbers greater than or equal to 0. Unlike the Poisson distribution, the variance and the mean are not equivalent. This suggests it might serve as a useful approximation for modeling counts with variability different from its mean. The variance of a negative binomial distribution is a function of its mean and has an additional parameter `k` called the dispersion parameter. Say our count is a random variable Y from a negative binomial distribution, when the variance of Y is:"

$$var(Y) = \mu + \mu^2 lk$$

"As the dispersion parameter gets larger and larger, the variance converges to the same value as the mean, and the negative binomial turns into a Poisson distribution."

```
## [1] "Mean of Target: 3"
```

```
## [1] "Variancea of Target: 3.711"
```

It appears that there is a slight overdispersion with the variance greater than the mean as shown above.

3.2.1 Negative binomial regression model 1

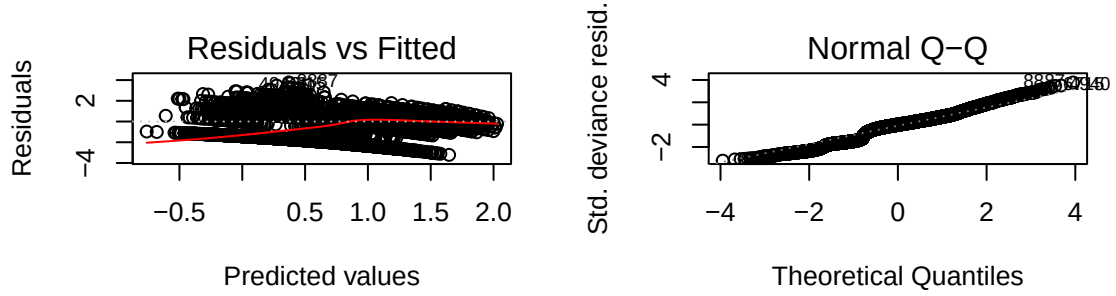
First negative binomial model is using all variables and selecting predictors using stepAIC.

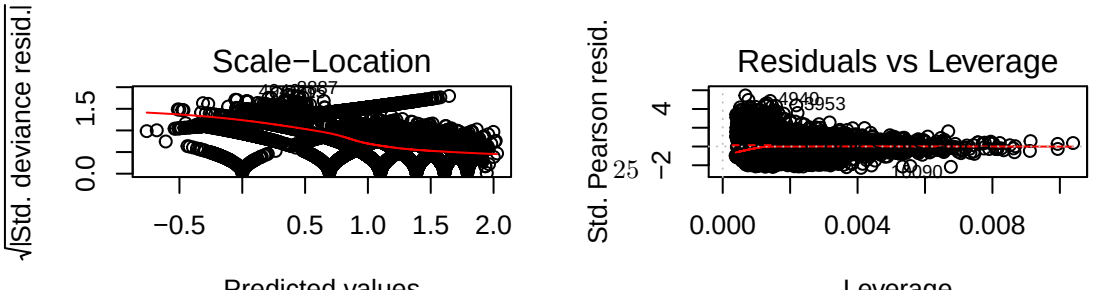
Observations	12795
Dependent variable	TARGET
Type	Generalized linear model
Family	Negative Binomial(40840)
Link	log

$\chi^2()$	0.50	0.16	45611.20	45775.25
Pseudo-R² (Cragg-Uhler)	0.50	0.16	45611.20	45775.25
Pseudo-R² (McFadden)	0.50	0.16	45611.20	45775.25
AIC	0.50	0.16	45611.20	45775.25
BIC	0.50	0.16	45611.20	45775.25

	Est.	S.E.	z val.	p
(Intercept)	0.70	0.20	3.53	0.00
Density	-0.27	0.19	-1.39	0.17
pH	-0.01	0.01	-1.59	0.11
Alcohol	0.00	0.00	2.83	0.00
LabelAppeal1	0.23	0.04	6.18	0.00
LabelAppeal2	0.42	0.04	11.47	0.00
LabelAppeal3	0.56	0.04	14.76	0.00
LabelAppeal4	0.69	0.04	16.36	0.00
AcidIndex	-0.08	0.00	-17.37	0.00
STARS1	0.77	0.02	39.18	0.00
STARS2	1.08	0.02	59.38	0.00
STARS3	1.20	0.02	62.65	0.00
STARS4	1.32	0.02	54.47	0.00
FixedAcidity	-0.00	0.00	-0.05	0.96
VolatileAcidity	-0.03	0.00	-5.28	0.00
CitricAcid	0.01	0.00	1.23	0.22
ResidualSugar	0.00	0.00	0.30	0.76
Chlorides	-0.03	0.01	-2.62	0.01
FreeSulfurDioxide	0.00	0.00	2.58	0.01
TotalSulfurDioxide	0.00	0.00	3.68	0.00
Sulphates	-0.01	0.00	-2.15	0.03

Standard errors: MLE





It is shown that there is an issue with non-constant variance and long tails in the qqplot. The best performing dataset was the one with negative values are arithmetically scaled from lower bound of IQR*1.5 to 0, and lesser values dropped. The negative binomial model appears to have many multiple statistically significant values.

3.2.2 Negative binomial regression model 2

Observations	12795
Dependent variable	TARGET
Type	Generalized linear model
Family	Negative Binomial(40838)
Link	log

$\chi^2()$	0.50	0.16	45606.79	45748.46
Pseudo-R^2 (Cragg-Uhler)	0.50	0.16	45606.79	45748.46
Pseudo-R^2 (McFadden)	0.50	0.16	45606.79	45748.46
AIC	0.50	0.16	45606.79	45748.46
BIC	0.50	0.16	45606.79	45748.46

	Est.	S.E.	z val.	p
(Intercept)	0.71	0.20	3.56	0.00
Density	-0.27	0.19	-1.40	0.16
pH	-0.01	0.01	-1.58	0.11
Alcohol	0.00	0.00	2.84	0.00
LabelAppeal1	0.23	0.04	6.18	0.00
LabelAppeal2	0.43	0.04	11.47	0.00
LabelAppeal3	0.56	0.04	14.77	0.00
LabelAppeal4	0.69	0.04	16.36	0.00
AcidIndex	-0.08	0.00	-17.53	0.00
STARS1	0.77	0.02	39.20	0.00
STARS2	1.08	0.02	59.42	0.00
STARS3	1.20	0.02	62.68	0.00
STARS4	1.32	0.02	54.50	0.00
VolatileAcidity	-0.03	0.00	-5.29	0.00
Chlorides	-0.03	0.01	-2.63	0.01
FreeSulfurDioxide	0.00	0.00	2.58	0.01
TotalSulfurDioxide	0.00	0.00	3.70	0.00
Sulphates	-0.01	0.00	-2.15	0.03

Standard errors: MLE

```
## $residual.deviance
## [1] 13624
##
## $residual.degrees.of.freedom
## [1] 12774
##
## $chisq.p.value
## [1] 0.000000094
```

```
## $residual.deviance
## [1] 13626
##
## $residual.degrees.of.freedom
## [1] 12777
##
## $chisq.p.value
## [1] 0.000000099
```

There is no significant improvement after using `stepAIC` to select the needed predictors. The reason to use Zero Dispersion Counts model is due to an inflated number of zeros in our counts target.

3.3 Linear regression models

3.3.1 Linear regression model 1

3.3.2 Linear regression model 2

Table 6: Comparison of models

model	BIC	Log-likelihood	Degrees of freedom
Poisson-Logit Hurdle Regression	40816.6060825119	-20280.6413861792	27
Zero-inflated Poisson	40914.6863017303	-20329.6814957884	27
Negative Binomial 1	45786.5052909723	-22789.2277382516	23
Negative Binomial 2	45775.2523193232	-22783.6012524271	22
Linear Model	43336.4446687359	-21616.2098807507	11
Linear Model 2	42836.2580131354	-21186.4371677271	49

4 SELECT MODELS

4.1 Comparison of models

Assessing the fit of a count regression model is not necessarily straightforward; often we just look at residuals, which invariably contain patterns of some form due to the discrete nature of the observations, or we plot observed versus fitted values as a scatter plot.

Kleiber and Zeileis (2016) https://arxiv.org/abs/1605.01311 proposes `rootogram` as an improved approach to the assessment of fit of a count regression model. The paper is illustrated using R and the authors' countreg package.

Rootograms are calculated using the rootogram() function. You can provide the observed and expected (given the model) counts as arguments to rootogram() or, most usefully for our purposes, a fitted count model object from which the relevant values will be extracted. rootogram() knows about glm, gam, gamlss, hurdle, and zeroinfl objects at the time of writing.

Three different kinds of rootograms are discussed in the paper

- Standing,
- Hanging, and
- Suspended.

Kleiber and Zeileis (2016) recommend hanging or suspended rootograms. Which type of rootogram is produced is controlled via argument style.

We will look at six different models, two poisson models, two negative-binomial models and an ols regression thrown in for good measure.

Both the Poisson-Logit Hurdle Regression and the zero-inflated poisson are very close in log likelihoods and BIC's.

The Poisson-Logit Hurdle Regression provides a closer fit to the observed than does the other models. The hurdle model is a modified count model in which there are two processes, one generating the zeros and one generating the positive values.

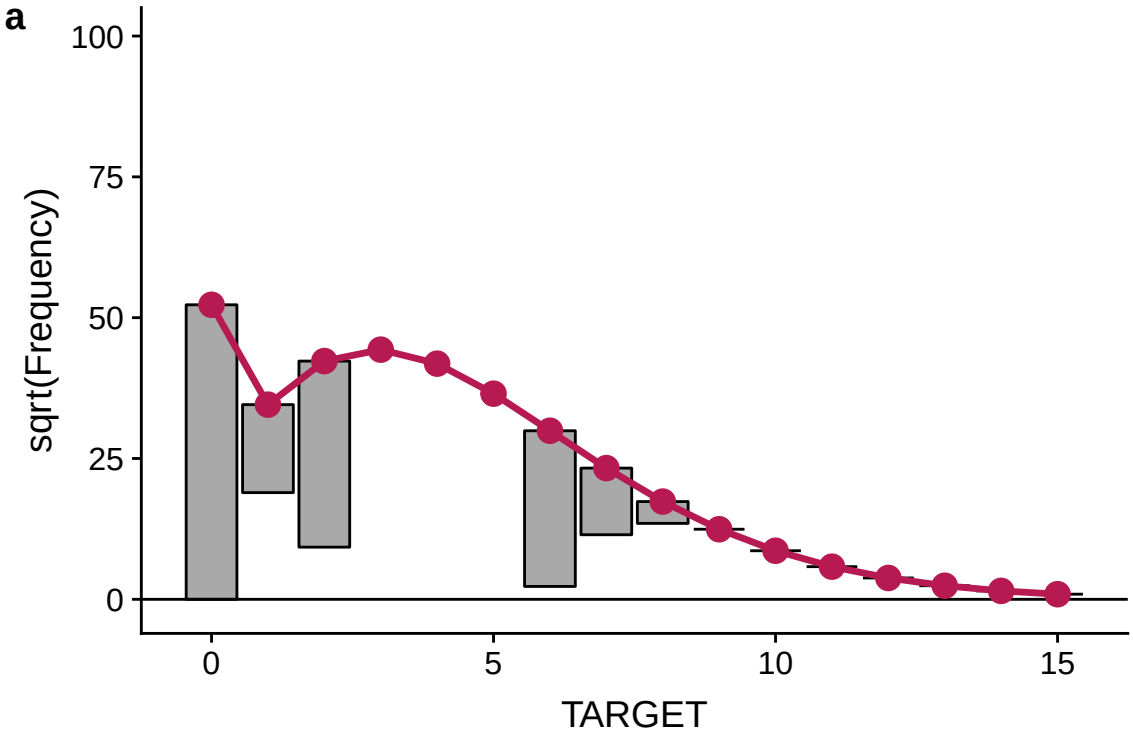
The Poisson-logit hurdle model is clearly the best choice here. The results for this model are given below.

```
##
## Call:
## hurdle(formula = TARGET ~ AcidIndex + Alcohol + LabelAppeal + STARS |
##     VolatileAcidity + FreeSulfurDioxide + TotalSulfurDioxide + pH +
##         Sulphates + Alcohol + LabelAppeal + AcidIndex + STARS, data = minusinfluential)
##
## Pearson residuals:
##      Min       1Q   Median       3Q      Max
## -2.09275 -0.44331 -0.00245  0.39568  4.54188
##
```

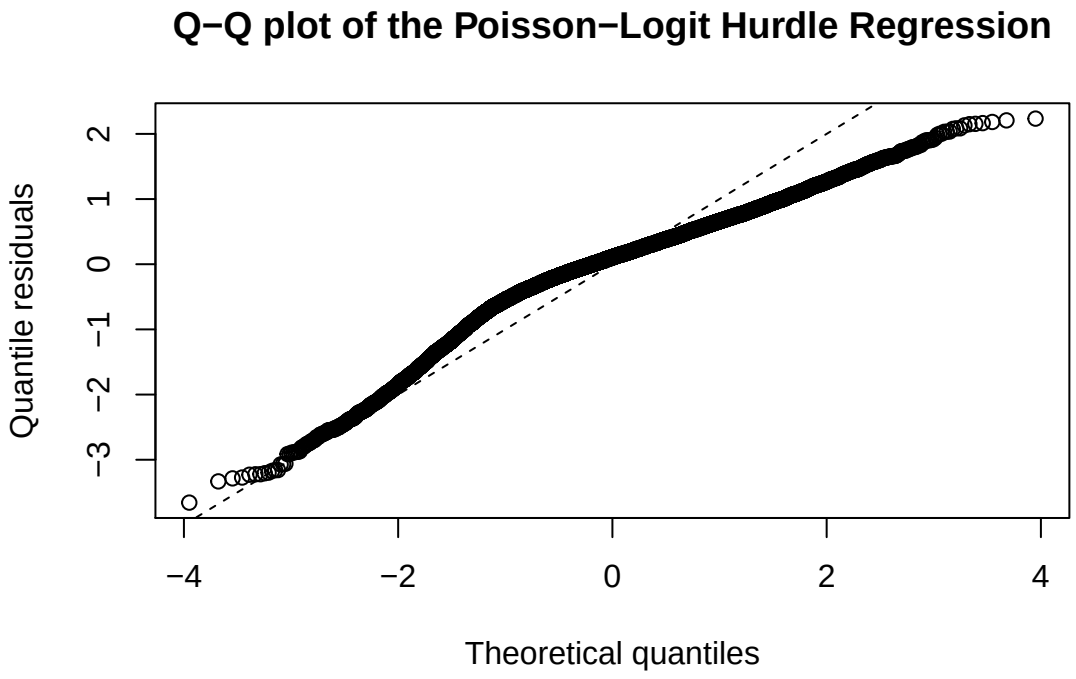
```
## Count model coefficients (truncated poisson with log link):
##               Estimate Std. Error z value             Pr(>|z|)
## (Intercept)    0.33863    0.06713    5.04           0.00000046 ***
## AcidIndex     -0.01705    0.00492   -3.46              0.00053 ***
## Alcohol        0.00732    0.00145    5.07           0.00000041 ***
## LabelAppeal-1  0.53977    0.04973   10.85 < 0.0000000000000002 ***
## LabelAppeal0   0.84299    0.04880   17.27 < 0.0000000000000002 ***
## LabelAppeal1   1.03971    0.04937   21.06 < 0.0000000000000002 ***
## LabelAppeal2   1.19915    0.05324   22.53 < 0.0000000000000002 ***
## STARS1         0.05342    0.02146    2.49              0.01283 *
## STARS2         0.16856    0.02002    8.42 < 0.0000000000000002 ***
## STARS3         0.25901    0.02098   12.35 < 0.0000000000000002 ***
## STARS4         0.36350    0.02592   14.02 < 0.0000000000000002 ***
## Zero hurdle model coefficients (binomial with logit link):
##                      Estimate Std. Error z value             Pr(>|z|)
## (Intercept)          4.433024   0.399013   11.11 < 0.0000000000000002 ***
## VolatileAcidity     -0.187381   0.036488   -5.14       0.000000281567 ***
## FreeSulfurDioxide    0.000666   0.000197    3.37              0.00074 ***
## TotalSulfurDioxide   0.000819   0.000127    6.46       0.000000000103 ***
## pH                  -0.172426   0.041928   -4.11       0.000039149953 ***
## Sulphates           -0.095360   0.030502   -3.13              0.00177 **
## Alcohol             -0.018903   0.007666   -2.47              0.01367 *
## LabelAppeal-1       -0.483545   0.137302   -3.52              0.00043 ***
## LabelAppeal0        -0.903071   0.134048   -6.74       0.000000000016 ***
## LabelAppeal1        -1.448751   0.143581  -10.09 < 0.0000000000000002 ***
## LabelAppeal2        -1.879326   0.223186   -8.42 < 0.0000000000000002 ***
## AcidIndex           -0.388161   0.021425  -18.12 < 0.0000000000000002 ***
## STARS1               1.831113   0.061349   29.85 < 0.0000000000000002 ***
## STARS2               4.268926   0.117246   36.41 < 0.0000000000000002 ***
## STARS3              20.250040 363.224872    0.06              0.95554
## STARS4              20.406402 695.238140    0.03              0.97658
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Number of iterations in BFGS optimization: 18
## Log-likelihood: -2.03e+04 on 27 Df
```

4.2 Diagnostic plots

Rootogram as an improved approach to the assessment of fit of a count regression model. Expected counts, given the model, are shown by the thick red line, and observed counts are shown as bars, which in a hanging rootogram are show hanging from the red line of expected count.



As a final check we can also look at the Q-Q plot of the quantile residuals in the hurdle model. These look fairly normal and show no suspicious departures from the model.



4.3 Prediction

We ran predictions on our final model and plotted the distribution next to the distribution from our target in the training data set to compare.

Table 7: TARGET Predictions

	n	min	mean	median	max	sd
X1	2523	0.11	3	3	6.8	1.4

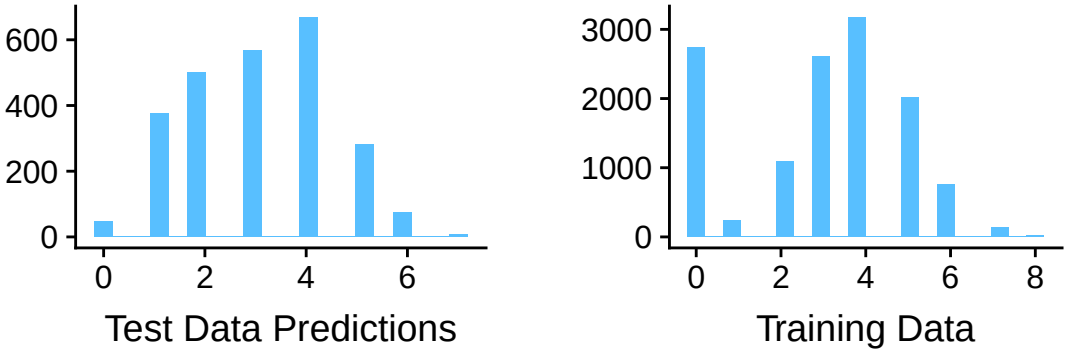


Figure 14: Predictions vs. training data

5 Appendix

The appendix is available as script.R file in `project5_wine` folder.

https://github.com/betsyrosalen/DATA_621_Business_Analyt_and_Data_Mining

```
# load data
train <- read.csv ('https://raw.githubusercontent.com/betsyrosalen/DATA_621_Business_Analyt_and_Data_Mi
                   stringsAsFactors = F, header = T)
test <- read.csv('https://raw.githubusercontent.com/betsyrosalen/DATA_621_Business_Analyt_and_Data_Minin
                 stringsAsFactors = F, header = T)

# remove index
train$INDEX <- NULL
test$INDEX <- NULL

vars <- rbind(c('TARGET','Number of Cases Purchased','count response'),
                 c('AcidIndex','Method of testing total acidity by using a weighted avg','continuous nu
                 c('Alcohol','Alcohol Content','continuous numerical predictor'),
                 c('Chlorides','Chloride content of wine','continuous numerical predictor'),
                 c('CitricAcid','Citric Acid Content','continuous numerical predictor'),
                 c('Density','Density of Wine','continuous numerical predictor'),
                 c('FixedAcidity','Fixed Acidity of Wine','continuous numerical predictor'),
                 c('FreeSulfurDioxide','Sulfur Dioxide content of wine','continuous numerical predictor
                 c('LabelAppeal','Marketing Score indicating the appeal of label design','categorical p
                 c('ResidualSugar','Residual Sugar of wine','continuous numerical predictor'),
                 c('STARS','Wine rating by a team of experts. 4 = Excellent, 1 = Poor','categorical pre
                 c('Sulphates','Sulfate conten of wine','continuous numerical predictor'),
                 c('TotalSulfurDioxide','Total Sulfur Dioxide of Wine','continuous numerical predictor'
                 c('VolatileAcidity','Volatile Acid content of wine','continuous numerical predictor'),
                 c('pH','pH of wine','continuous numerical predictor') )

colnames(vars) <- c('VARIABLE','DEFINITION','TYPE')


# Summary Statistics

train_num <- train[, c( 'AcidIndex','Alcohol', 'Density',
                           'Sulphates', 'pH', 'TotalSulfurDioxide','FreeSulfurDioxide', 'Chlorides',
                           'ResidualSugar', 'CitricAcid', 'VolatileAcidity','FixedAcidity', 'TARGET')]

train_cat <- train[, c('STARS', 'LabelAppeal')]
train_cat$STARS <- as.factor(train_cat$STARS )
train_cat$LabelAppeal <- as.factor(train_cat$LabelAppeal )

train_num_stats <- describe(train_num)[,c(2,8,3,5,9,4)]
train_cat_stats <- summary(train_cat[, c('STARS', 'LabelAppeal')])

# Data Distribution
hist <- train %>%
  gather() %>%
  ggplot(aes(value)) +
  facet_wrap(~ key, scales = "free") +
  geom_histogram(fill = "#58BFFF") +
  xlab("") +
```

```
  ylab("") +
  theme(panel.background = element_blank())

h.train <- train
h.train$STARS[is.na(h.train$STARS)] <- 0
cbPalette <- c("#58BFFF", "#3300FF", "#E69F00", "#009E73", "#CC79A7")
hist.STARS <- h.train[, c('TARGET', 'STARS')] %>%
  gather(-STARS, key = "var", value = "val") %>%
  ggplot(aes(x = val, fill=factor(STARS))) +
  geom_bar(alpha=0.8) +
  facet_wrap(~ STARS, scales = "free") +
  scale_fill_manual("STARS", values = cbPalette) +
  xlab("TARGET") +
  ylab("") +
  theme(panel.background = element_blank(), legend.position="top")
hist.STARS

hist.LabelAppeal <- h.train[, c('TARGET', 'LabelAppeal')] %>%
  gather(-LabelAppeal, key = "var", value = "val") %>%
  ggplot(aes(x = val, fill=factor(LabelAppeal))) +
  geom_bar(alpha=0.8) +
  facet_wrap(~ LabelAppeal, scales = "free") +
  scale_fill_manual("LabelAppeal", values = cbPalette) +
  xlab("TARGET") +
  ylab("") +
  theme(panel.background = element_blank(), legend.position="top")
hist.LabelAppeal
#c("#58BFFF", "#3300FF")

# Boxplot

scaled.train.num <- as.data.table(scale(train[, c('AcidIndex','Alcohol', 'Density',
                                                  'Sulphates', 'pH', 'TotalSulfurDioxide','FreeSulfurDi
                                                  'ResidualSugar', 'CitricAcid', 'VolatileAcidity','Fix


melt.train <- melt(scaled.train.num)

scaled_boxplots <- ggplot(melt.train, aes(variable, value)) +
  geom_boxplot(width=.5, fill="#58BFFF", outlier.colour="red", outlier.size = 1) +
  stat_summary(aes(colour="mean"), fun.y=mean, geom="point",
               size=2, show.legend=TRUE) +
  stat_summary(aes(colour="median"), fun.y=median, geom="point",
               size=2, show.legend=TRUE) +
  coord_flip() +
  #scale_y_continuous(labels = scales::comma,
  #                   breaks = seq(0, 110, by = 10)) +
  labs(colour="Statistics", x="", y="") +
  scale_colour_manual(values=c("#9900FF", "#3300FF")) +
  theme(panel.background=element_blank(), legend.position="top")
scaled_boxplots

#Scatter plot between numeric predictors and the TARGET
linearity <- train %>%
```

```
  gather(-TARGET, key = "var", value = "value") %>%
  ggplot(aes(x = value, y = TARGET)) +
  geom_point(alpha=0.1) +
  stat_smooth() +
  facet_wrap(~ var, scales = "free", ncol=3) +
  ylab("TARGET") +
  xlab("") +
  theme(panel.background = element_blank())

#Scatter plot between log transformed predictors and the log transformed TARGET filtered for rows where
logged_vals <- train[,c('AcidIndex','Alcohol', 'Density',
                        'Sulphates', 'pH', 'TotalSulfurDioxide','FreeSulfurDioxide', 'Chlorides',
                        'ResidualSugar', 'CitricAcid', 'VolatileAcidity','FixedAcidity', 'TARGET')]
logged_vals <- logged_vals %>%
  filter(TARGET>0) %>%
  log()

linearity_log <- logged_vals %>%
  gather(-TARGET, key = "var", value = "value") %>%
  ggplot(aes(x = value, y = TARGET)) +
  geom_point(alpha=0.1) +
  stat_smooth() +
  facet_wrap(~ var, scales = "free", ncol=3) +
  ylab("TARGET") +
  xlab("") +
  theme(panel.background = element_blank())

#Box Cox
bc <- train[train[, 'TARGET'] > 0, ]



## Square Root Transformed Predictors and Log transformed Target Linearity Plot
X <- train[train[, 'TARGET']>0,
             c('AcidIndex','Alcohol', 'Density',
               'Sulphates', 'pH', 'TotalSulfurDioxide','FreeSulfurDioxide', 'Chlorides',
               'ResidualSugar', 'CitricAcid', 'VolatileAcidity','FixedAcidity')]
sqroot_vals <- data.table(cbind(log(train[train[, 'TARGET']>0,'TARGET']),
                                sapply(X, sqrt)))

colnames(sqroot_vals)[1] <- 'TARGET'

linearity_root_2 <- sqroot_vals %>%
  gather(-TARGET, key = "var", value = "value") %>%
  ggplot(aes(x = value, y = TARGET)) +
  geom_point(alpha=0.1) +
  stat_smooth() +
  facet_wrap(~ var, scales = "free", ncol=3) +
  ylab("TARGET") +
  xlab("") +
  theme(panel.background = element_blank())


# DATA PREPARATION <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
```

```
# STARS NA<- 0
train$STARS[is.na(train$STARS)] <- 0
test$STARS[is.na(test$STARS)] <- 0
test$STARS <- as.factor(test$STARS)
test$LabelAppeal <- as.factor(test$LabelAppeal)

##1. Data as is : train_imputed
# Then we run imputation
# impute NAs using MICE for all variables with exception of STARS
train_mice <- mice::mice(train, m = 2, method='cart', maxit = 2, print = FALSE)
train_imputed <- mice::complete(train_mice)

train_imputed_raw <- train_imputed

train_imputed$STARS <- as.factor(train_imputed$STARS)
train_imputed$LabelAppeal <- as.factor(train_imputed$LabelAppeal)
#-------------------------------------------------------------------------------------------
##2. Data by shifted by min value:

train_plusmin <- train_imputed_raw

# list of columns that will be transformed
cols <- c("FixedAcidity","VolatileAcidity",
          "CitricAcid","ResidualSugar",
          "Chlorides","FreeSulfurDioxide",
          "TotalSulfurDioxide","Sulphates","Alcohol")

# Transformation of train_plusmin by adding the minimum value plus one
for (col in cols) {
  train_plusmin[, col] <- train_plusmin[, col] + abs(min(train_plusmin[, col])) + 1
}

train_plusmin$STARS <- as.factor(train_plusmin$STARS)
train_plusmin$LabelAppeal <- as.factor(train_plusmin$LabelAppeal)
#-------------------------------------------------------------------------------------------
##3. Data by Jeremy's method
# arithmetically scaled from lower bound of IQR*1.5 to 0, and lesser values dropped: train_minscaled
# Subset variables with values for frequencies / concentrations / amounts that are < 0
train_scaling_subset <- train_imputed_raw %>%
  dplyr::select(FixedAcidity,
                VolatileAcidity,
                CitricAcid,
                ResidualSugar,
                Chlorides,
                FreeSulfurDioxide,
                TotalSulfurDioxide,
                Sulphates)
# dplyr::rename_all(paste0, '_scaled')

# Function to additively scale values by amount equivalent to lower bound of 1.5 * IQR
# then drop anything below 0 and leaves NAs as they are
positive_scale <- function(x) {
  low_bound <- mean(x, na.rm = TRUE) - (stats::IQR(x, na.rm = TRUE) * .5) * 1.5
```

```
  if(is.na(x)) {
    x = NA
  } else if(x < low_bound) {
    x = 0
  } else {
    x = x + abs(low_bound)
  }
}

# Rescale subset of variables with values < 0
train_iqrscaled_subset <- lapply(train_scaling_subset,
                                 FUN = function(x) sapply(x, FUN = positive_scale)) %>%
  as.data.frame()

# Join scaled subset back to other variables
train_plusiqr15 <- train_imputed_raw %>%
  dplyr::select(TARGET,
                Density,
                pH,
                Alcohol,
                LabelAppeal,
                AcidIndex,
                STARS) %>%
  cbind(train_iqrscaled_subset)

# Rescale discrete label appeal variable and factorize
train_plusiqr15$LabelAppeal <- train_imputed_raw %>%
  select(LabelAppeal) %>%
  sapply(FUN = function(x) x + 2) %>%
  as.factor()

train_plusiqr15$STARS <- as.factor(train_plusiqr15$STARS)
#-------------------------------------------------------------------------------------------
##4. Data by ABS and Log

# Convert subset of variables to absolute value
train_scaling_subset2 <- train_imputed_raw %>%
  dplyr::select(FixedAcidity,
                VolatileAcidity,
                CitricAcid,
                ResidualSugar,
                Chlorides,
                FreeSulfurDioxide,
                TotalSulfurDioxide,
                Sulphates,
                Alcohol)

train_absscaled_subset <- lapply(train_scaling_subset2,
                                 FUN = function(x) sapply(x, FUN = abs)) %>%
  as.data.frame()

# lapply(train_absscaled_subset, min)

# Join absolute value-scaled subset back to other continuous variables
```

```
train_abs <- train_imputed_raw %>%
  dplyr::select(Density,
                pH,
                AcidIndex) %>%
  cbind(train_absscaled_subset)

# Log-scale all continuous variables, adding constant of 1
train_abslog <- lapply(train_abs, FUN = function(x)
  sapply(x, FUN = function(x) log(x+1))) %>%
  as.data.frame()

# Rescale discrete label appeal variable and factorize
train_abslog$LabelAppeal <- train_imputed_raw %>%
  select(LabelAppeal) %>%
  sapply(function(x) x + 2) %>%
  as.factor()

# Map remaining variables to dataframe
#train_abslog$INDEX <- train_imputed$INDEX
train_abslog$TARGET <- train_imputed_raw$TARGET
train_abslog$STARS <- train_imputed_raw$STARS
train_abslog$STARS <- as.factor(train_abslog$STARS)
```